# Introduction to Quantum Information, Computation and Communication PHYS 345 Spring 2024

Steven M. Girvin

Disclosure:
The author is a shareholder in IBM Corporation,
and consultant for, and equity holder in,
Quantum Circuits, Inc., a quantum computing company.

[Compiled: May 4, 2024]

## 0.1 Preface

These lecture notes are intended for undergraduates from a variety of disciplines (e.g. physics, mathematics, chemistry, computer science, electrical engineering) interested in an initial introduction to the new field of quantum information science.

The reader may wish to consult texts such as 'An Introduction to Quantum Computing,' by Phillip Kaye, Raymond Laflamme and Michele Mosca, hereafter referred to as [KLM], or 'Quantum Computing: A Gentle Introduction,' by Eleanor Rieffel and Wolfgang Polak, hereafter referred to as [RP]. Computer scientists may be interested in consulting 'Quantum Computer Systems,' by Yongshan Ding and Fredrick Chong. Having mastered the lecture notes for this class, you will be ready to open the bible in the field, 'Quantum Computation and Quantum Information,' by Michael Nielsen and Isaac Chuang, universally referred to as 'Mike and Ike' (perhaps after the candy bar of the same name).

# Contents

0.1 Preface . . . . . . . . . . i

**1 Introduction** **1**
1.1 What is Information? . . . . . . . . . . 2
1.2 Error Correction and Data Compression . . . . . . . . . . 14
1.3 What is a computation? . . . . . . . . . . 23
1.4 Universal Gate Sets . . . . . . . . . . 30

**2 Quantum Bits: Qubits** **36**
2.1 Quantum Bits for Computer Scientists . . . . . . . . . . 37
2.2 Quantum Bits for Budding Physicists . . . . . . . . . . 41

**3 Introduction to Hilbert Space** **57**
3.1 Linear Operators on Hilbert Space . . . . . . . . . . 60
3.2 Dirac Notation for Operators . . . . . . . . . . 67
3.3 Orthonormal bases for qubit states . . . . . . . . . . 68
3.4 Rotations in Hilbert Space . . . . . . . . . . 74
3.5 Hilbert Space and Operators for Multiple Qubits . . . . . . . . . . 86

**4 Two-Qubit Gates and Multi-Qubit Entanglement** **95**
4.1 Introduction: Multi-Qubit Operations and Measurements . . 95
4.2 Multi-Qubit Measurements . . . . . . . . . . 96
4.3 Two-Qubit Gates . . . . . . . . . . 99
4.4 Bell Inequalities . . . . . . . . . . 111
4.5 Quantum Dense Coding . . . . . . . . . . 116
4.6 No-Cloning Theorem Revisited . . . . . . . . . . 122
4.7 Quantum Teleportation . . . . . . . . . . 126
4.8 YET TO DO: . . . . . . . . . . 129

**5 Algorithms and Complexity Classes 131**
5.1 Phase 'Kickback' of a Controlled-Unitary Operation . . . . . 132
5.2 Exponentiation Gadget . . . . . . . . . . . . . . . . . . . . . . . 135
5.3 Quantum Oracles . . . . . . . . . . . . . . . . . . . . . . . . . 138
5.4 Deutsch Algorithm . . . . . . . . . . . . . . . . . . . . . . . . 142
5.5 Deutsch-Jozsa Algorithm . . . . . . . . . . . . . . . . . . . . . 146
5.6 Bernstein-Vazirani Problem . . . . . . . . . . . . . . . . . . . 152
5.7 Simon's Algorithm . . . . . . . . . . . . . . . . . . . . . . . . 159
5.8 Grover Search Algorithm . . . . . . . . . . . . . . . . . . . . . 170
5.9 VQE and QAOA Algorithms . . . . . . . . . . . . . . . . . . . 181
5.10 Phase Estimation Algorithm . . . . . . . . . . . . . . . . . . . 181
5.11 Quantum Fourier Transform . . . . . . . . . . . . . . . . . . . 181

**6 Quantum Error Correction 182**
6.1 An advanced topic for the experts . . . . . . . . . . . . . . . . 188

**7 Yet To Do 191**

**A Quick Review of Probability and Statistics 195**
A.1 Randomness . . . . . . . . . . . . . . . . . . . . . . . . . . . 195
A.2 Probabilities . . . . . . . . . . . . . . . . . . . . . . . . . . . 196
A.3 Statistical Estimators . . . . . . . . . . . . . . . . . . . . . . . 200
A.4 Joint Probability Distributions . . . . . . . . . . . . . . . . . . 207

**B Formal Definition of a Vector Space 211**
B.1 Basis Vectors and the Dimension of a Vector Space . . . . . 212
B.2 Inner Products and Norms . . . . . . . . . . . . . . . . . . . . 213
B.3 Dirac Notation, Outer Products and Operators for Single and Multiple Qubits . . . . . . . . . . . . . . . . . . . . . . . . 215
B.4 The Vector Space of Bit Strings . . . . . . . . . . . . . . . . . 220

**C Handy Mathematical Identities 222**

**D Suggestions for Projects 226**
D.1 Gates and Instruction Sets . . . . . . . . . . . . . . . . . . . . 226
D.2 Efficient Representation of Quantum States . . . . . . . . . . 227
D.3 Algorithms . . . . . . . . . . . . . . . . . . . . . . . . . . . . 228
D.4 Quantum Error Correction and Fault Tolerance . . . . . . . 229
D.5 Quantum Communication and Security . . . . . . . . . . . . 230

D.6 Quantum Complexity Classes . . . . . . . . . . . . . . . . . . 230
D.7 Quantum Hardware Platforms . . . . . . . . . . . . . . . . . . . 230

# Chapter 1

# Introduction

The first quantum revolution began with the development of the quantum theory early in the 20th century. Scientists carrying out fundamental, curiosity-driven research about the nature of reality, discovered that reality is not what it seems to humans. The laws of classical mechanics developed by Galileo, Newton and others beautifully and accurately describe the motion of macroscopic objects like baseballs, soccer balls and planets, but they fail for microscopic objects like atoms and electrons. This research was not motivated by practical applications and indeed seemed likely to have none. And yet it did. The first quantum revolution led to the invention of the transistor, the laser and the atomic clock. These three inventions in turn produced the fantastic technological revolution of the second half of the 20th century that brought us (among many other things) the computer, fiber-optic communication and the global positioning system.

There is now a second quantum revolution underway in the 21st century. It is not based on a new theory of quantum mechanics. Indeed the quantum theory is essentially unchanged from the 1920's and is the most successful theory in all of physical science–it makes spectacularly accurate quantitative predictions. Instead this second revolution is based on a new understanding of the often counter-intuitive aspects of the quantum theory. We now understand that the quantum devices invented in the 20th century do not use the full power that is available to quantum machines. This new understanding has rocked the foundations of of theoretical computer science laid down nearly a century ago by Church and Turing in their exploration of the concept of computability. This new understanding is leading to the invention of radically new techniques for processing and communicating information and for making measurements of the tiniest signals. These new techniques are being rapidly developed and if they can be transformed into practical technologies (and this is a big 'if'), the second quantum revolution could have an impact on human society at least as profound as the technological revolution of the 20th century.

Achieving the full potential of the second quantum revolution will require interdisciplinary research involving physics, computer science, electrical engineering, mathematics and many other disciplines. My goal is to present quantum information, computation and communication in a way that is accessible to undergraduates from any quantitative discipline. This will require removing a lot of the physics language, historical background and motivation to the quantum theory and simply presenting a set of rules by which one can describe the states and operations of quantum machines.

Before entering the quantum realm, let us begin by reviewing some basic concepts about information and how it is quantified and processed. By analogy with the fact that the pre-quantum theory of mechanics is known today as 'classical' mechanics, we will refer to the pre-quantum theory of information as classical information.

Both classical and quantum information involves the concept of randomness. Before going further, the reader is therefore urged to review the basic concepts of probability and statistics in Appendix A

## 1.1 What is Information?

Information is 'news' or 'surprise.' If Alice tells Bob[1] something he already knows, she has not transmitted any information to him and Bob has not learned (received) any information. Suppose for example that Bob asks Alice a question that has two possible answers (yes/no or true/false, say). Further suppose that Bob does not know the answer to his own question. Alice can transmit the answer as a message to Bob by choosing one of two (agreed upon) symbols, say T or F for true or false, or y or n for yes or no. For simplicity we will assume that Alice transmits a 'binary digit,' or 'bit', either the symbol 0 or 1.

[1]In the quantum communication literature, it is traditional to refer to the two communicating parties as 'Alice' and 'Bob.' An eavesdropper listening in on the conversation is traditionally referred to as 'Eve.' Who says physicists don't have a sense of humor?

The word 'bit' is short for binary digit a number whose value can be represented by 0 or 1 in the binary numbering system (base 2 numbers). The word bit is also used to mean the amount of information contained in the answer to a yes/no or true/false question (assuming you have no prior knowledge of the answer and that the two answers are equally likely as far as you know). If Alice gives Bob either a 0 or a 1 drawn randomly with equal probability, then Bob receives one bit of information. Bits and base 2 numbers are very natural to use because information is *physical*. It is transmitted and stored using the states of physical objects, as illustrated in Fig. 1.1. For example an electrical switch can be open or closed and its state naturally encodes one 1 bit of information. Similarly a transistor in a computer chip can be in one of two electrical states, on or off. Discrete voltage levels in a computer circuit, say 0 volts and 5 volts are also used to encode the value of a bit. This discretization is convenient because it helps make the system robust against noise. Any value of voltage near 0 is interpreted as representing the symbol 0 and any value near 5 volts is interpreted as representing the symbol 1. Information can also be stored in small domains of magnetism in disk drives. Each magnetic domain acts like a small bar magnet whose magnetic moment (a vector quantity pointing in the direction running from the south pole to the north pole of the bar magnet) can only point up or down. Ordinarily a bar magnet can point in any direction in space, but the material properties of the disk drive are intentionally chosen to be anisotropic so that only two directions are possible. Information is also communicated via the states of physical objects. A light bulb can be on or off and the particles of light (photons) that it emits can be present or absent, allowing a distant observer to receive information.

**Box 1.1. Two meanings of 'bit'** The word bit can refer to the mathematical measure of information content of a message consisting of a string of symbols with a specified probability distribution. It can also refer to the physical object storing the information, as in 'My computer has 100 bytes of memory', with one byte being 8 bits. A useful table of prefixes is

| | | | |
|---|---|---|---|
| kilo | $10^3$ | peta | $10^{15}$ |
| mega | $10^6$ | exa | $10^{18}$ |
| giga | $10^9$ | zetta | $10^{21}$ |
| terra | $10^{12}$ | yotta | $10^{24}$ |

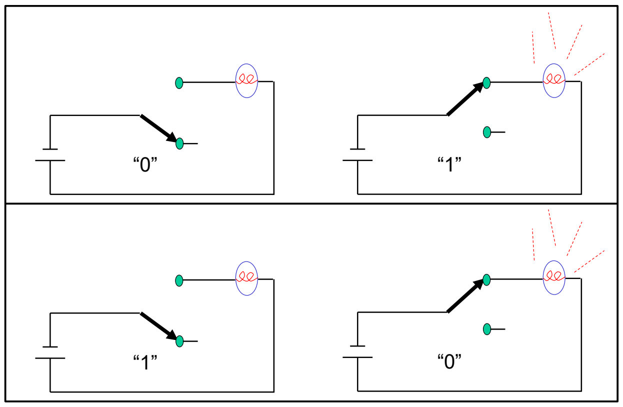

Figure 1.1: The fact that ‘information is physical’ is illustrated by this electrical circuit encoding one bit of information. Upper panel: Bit value 0 (1) is represented by the switch being open (closed) and the light bulb is off (on). Lower panel: There is only one other possible encoding of the classical information, namely the one in which states 0 and 1 and interchanged. A simple NOT operation transforms one encoding into the other. We will see that the quantum situation is much richer than this.

A storage register holding $N$ bits can be in *one* of $2^N$ distinct states. For example 3 classical bits can be in 8 states:

$$(000), (001), (010), (011), (100), (101), (110), (111),$$

corresponding to the 8 binary numbers whose (base 10) values run from 0 to 7. This is a remarkably efficient encoding–with an ‘alphabet’ of only two symbols (0 and 1) Alice can send Bob one of $2^{32} = 4,294,967,296$ possible messages at a cost of only having to transmit 32 bits (or 4 ‘bytes’). In fact, if all the messages are equally likely to be chosen, this is provably the most efficient possible encoding.[2]

Let us now turn this picture around by considering the same situation in which Alice sends one message from a collection of $M = 2^N$ possible distinct messages to Bob. If Alice is equally likely to choose all messages, this will maximize Bob’s ‘surprise’ because no information in advance about which message is likely to be sent. Thus the information he receives is maximized. We have seen that Alice must pay the price of sending $N = \log_2 M$ physical

[2]More precisely, there is no encoding more efficient than this. There are many encodings that are as efficient that differ simply by permutation of the symbols. For example, we could order the binary digits with the least significant to the left instead of to the right.

bits to Bob. This strongly suggests that we should quantify the amount of information that Bob receives as

$$H = \log_2 M = N \tag{1.1}$$

bits of information (one for each physical bit received). If the messages are not all equally likely to be sent, the surprise is less and we will see shortly how to modify this formula to account for that.

**Box 1.2. Reminder About Logarithms** In Eq. (1.1), $\log_2$ means logarithm base 2. Let us remind ourselves about some basic properties of logarithms in different bases:

$$\begin{aligned}
x &= e^{\ln x} && (1.2)\\
x &= 10^{\log_{10} x} = \left[e^{(\ln 10)}\right]^{(\log_{10} x)} = e^{(\ln 10)(\log_{10} x)} && (1.3)\\
x &= 2^{\log_2 x} = \left[e^{(\ln 2)}\right]^{(\log_2 x)} = e^{(\ln 2)(\log_2 x)} && (1.4)\\
\ln x &= (\ln 2)(\log_2 x) = (\ln 10)(\log_{10} x) && (1.5)\\
\log_2 x &= \left[\frac{1}{\ln 2}\right] \ln x \approx 1.442695 \ln x && (1.6)\\
\log_2 x &= \left[\frac{\ln 10}{\ln 2}\right] \log_{10} x \approx 3.32193 \log_{10} x, && (1.7)
\end{aligned}$$

where ln is the natural logarithm (log base $e$). Thus the choice of different bases simply multiplies the information measure by a constant factor. The standard choice of log base 2 gives us (by definition) the information content in bits.

Does it make sense that the information is logarithmic in the total number of possible messages? Indeed it does, because if Alice randomly chooses $m$ messages to send, the total number of possible compound messages is $M' = M^m$ and we have

$$H = \log_2 M^m = mN. \tag{1.8}$$

Thus the amount of information Bob receives is linear in the number of messages Alice sends, a natural and sensible property for any measure of information to have. It also makes sense from the number of physical bits Alice needs to send Bob. If sending one message costs $\log_2 M$ physical bits (when using the optimal encoding), then sending $m$ messages should require $m \log_2 M$ physical bits. As noted above, if we had chosen a different base for the logarithm, our measure of information would just change by a fixed scale factor. Only when we use base 2 is the information content measured in bits.

**Box 1.3. A Natural Measure of Information** A natural measure of the information content of a message randomly selected from a range of possibilities is the length of the shortest possible encoding of the messages. The intuition is that to transmit more information requires transmitting more symbols. Suppose that there is a choice of $M = 2^N$ possible messages (labeled say by a message number $j \in \{0, M-1\}$), and both Alice and Bob have a book that contains the messages written out in plain text. Each message could have a different length of text ranging from a single word to an entire book chapter. The only thing that Alice needs to send Bob is the number labeling the message, not the text of the message itself. The most efficient encoding is for Alice to send a string of $N$ bits that constitute the binary representation of the message number $j$. This measure of information content in terms of most efficient possible encoding is thus consisten with our definition in Eq. (1.1). Note that there are plenty of less efficient encodings. For example, Alice could transmit a string of $2^N$ bits, all of which are 0 except for the bit at position $j$ in the string. This would uniquely identify which message was sent but is exponentially less efficient than using the binary number representation which requires only transmitting a string of length $N$. Table 1.1 illustrates these two encodings for a set of $M = 8$ possible messages.

So far we have only considered the case where Alice chooses randomly with equal probability amongst $M = 2^N$ messages. It is the randomness of her choice that insures that Bob is always surprised by the message he receives. He has no way of guessing in advance what the message will be. You might wonder why Alice wants to send Bob random messages. If she were sending Bob text written in English or Swedish say, the characters she sends would not be random. For example, in English text the letters e,t and s occur more frequently than other letters. Perhaps Alice is a spy and she is sending secretly encoded messages to Bob. If Alice made the mistake of using a simple substitution cipher in which (say) e is always represented as g, t as w and s as p, etc., then an eavesdropper would notice that the symbols

| Message Number | Binary Encoding | Unary Encoding |
|---|---|---|
| 0 | 000 | 00000001 |
| 1 | 001 | 00000010 |
| 2 | 010 | 00000100 |
| 3 | 011 | 00001000 |
| 4 | 100 | 00010000 |
| 5 | 101 | 00100000 |
| 6 | 110 | 01000000 |
| 7 | 111 | 10000000 |

Table 1.1: Alice wants to transmit one of 8 messages to Bob. Since she and Bob each have a book containing the messages, she need only transmit the number labeling the message she wants to convey to Bob. The binary encoding requires only three bits. This is the shortest possible encoding (using two symbols) and thus a natural measure of the information content is 3 bits. There are many less efficient encodings such as the unary encoding shown in the third column of the table. Notice that this inefficient encoding is not very random since 0 occurs exponentially more frequently than 1 (relative to the binary encoding). It is this lack of randomness ('surprise') that makes the encoding inefficient.

g, w and p appear more frequently than others and this would give a clue that could be used to help break the code. To prevent this, Alice should instead use a code with the property that the encoded messages appear to be completely random. Perhaps Alice is not a spy, but merely wants to compress the data she sends so that the message is as short as possible. As we will see below, the non-randomness of natural language texts means that they can be compressed into shorter messages, but since the total amount of information must be preserved, the compressed messages are necessarily more random.

We already saw an illustration of this concept in the example given in Table 1.1. Provided that Alice selects her messages to send randomly with equal probability, then the bits in the binary code (middle column) are random and unpredictable, whereas the bits in the inefficient code are much less random (being mostly zeros). Suppose however that the probability distribution from which Alice selects her messages is not uniform. Say for example, message 0 is selected 93% of the time and the seven remaining messages are selected with probability 1%. Then the binary bits of the binary code are less random, being all zeros 93% of the time (since message 0 is encoded as (000) in the simple binary code). How do we quantify the information content in this case? It must be smaller because there is less surprise for Bob. This means that we should be able to find an even more efficient code to transmit the messages with fewer but more random bits.

In 1948, Claude Shannon, a scientist at Bell Telephone Laboratories, invented information theory and explained how to quantify information for messages drawn from a given probability distribution. He called the quantity of information the 'information entropy' and gave it the symbol $H$. Entropy is also a term from the field of thermodynamics[3] and is a measure of the randomness or unpredictability of the microscopic state of the molecules in a gas or fluid. Thermodynamic entropy can be viewed as the amount of information Bob gains when Alice tells him the microstate of the gas or fluid (i.e. the positions and velocities of all the particles). In a crystalline solid the positions of the atoms are highly ordered since they are stacked in an infinitely repeating pattern as shown in Fig. 1.2. Once you know the positions a few atoms you can predict the positions of the others. This is not true in a liquid which is much more disordered. Hence a liquid has greater thermodynamic entropy than a crystalline solid. Similarly the Shannon entropy of the message listing the positions of all the atoms in a liquid is much greater than that of the message listing all the positions of the atoms in a solid.

Shannon showed that $H$ is a simple function (which we derive in Box 1.5) of the probability of receiving different messages

$$H_M = -\sum_{j=1}^{M} p_j \log_2 p_j, \tag{1.9}$$

where $p_j$ is the probability of receiving the $j$th message from the list of $M$ distinct possible messages. We need the minus sign in the definition because probabilities always lie in the domain $0 \leq p \leq 1$ and so their log is always negative.[4] For the case we have been considering up to now, all messages

[3]Thermodynamics is the study of heat, work and phase transitions such as melting of solids into liquids and boiling of liquids into vapors. Chemists use it to predict chemical reactions and physicist and engineers use it to predict the efficiency of heat engines. In these fields, the standard symbol for entropy is $S$ and it is usually define with natural logarithms rather than base 2 logs.

[4]Note that the logarithm diverges as $p_j$ goes to zero, but only very slowly. Using l'Hôpital's rule, one sees that the limit $\lim_{p_j \to 0} p_j \log_2(p_j) = 0$. We implicitly use this smooth limit in the case where any of the probabilities actually are zero and the corresponding logarithm is undefined.

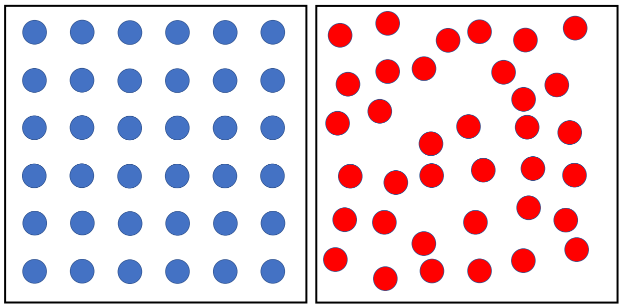

Figure 1.2: Left panel: positions of atoms in an ideal crystal. If you are told the positions of a few atoms in the upper left corner, you can predict the positions of the rest because of the simple repeating nature of the pattern. Right panel: snapshot of the positions of the atoms in a liquid at some moment. The positions are not completely random (e.g., no two atoms occupy the same space) but they are much more random than in the crystal. Thus the thermodynamic entropy is larger for the liquid, as is the Shannon information entropy of the message listing the positions of all the atoms.

have been equally likely, so $p_j = 1/M$ and

$$H_M = -\sum_{j=1}^{M} \frac{1}{M} \log_2 \frac{1}{M} = \log_2 M, \tag{1.10}$$

in agreement with our previous result in Eq. (1.1). We begin to see here that the information content of a set of messages is a function of the probability distribution over the set of possible messages.

It turns out that Shannon's Eq. (1.9) is still correct even if different messages have different probabilities. Indeed we should think of Shannon entropy as a general property of any probability distribution. The entropy is a measure of the randomness associated with sampling from the distribution. To begin to understand why this is so, consider the case $M = 2$ with the probability of receiving the symbol 0 being $p_0$ and the probability of receiving the symbol 1 being $p_1 = 1 - p_0$ The Shannon entropy for this special case, $H_2 = h(p_0, p_1)$ is plotted in Fig. 1.3. The graph makes intuitive sense because the information content of the message goes to zero when the probability of either symbol approaches unity and the other approaches zero. In this limit the messages are completely predictable and there is no surprise left. We also see that, consistent with our earlier intuition above, when the messages are equally likely ($p_0 = p_1 = \frac{1}{2}$), the randomness is maximized and the information content of the message takes on its maximum value of precisely

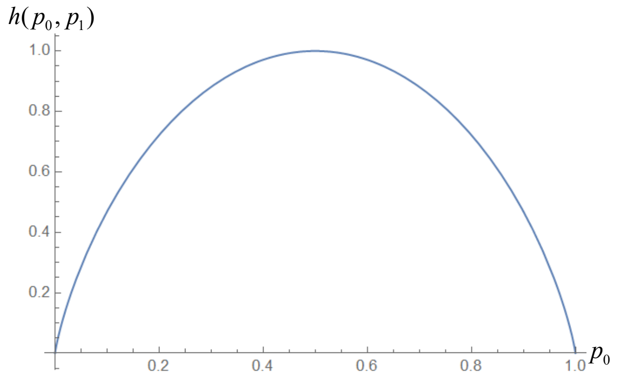

Figure 1.3: Shannon information entropy for a message containing a single symbol. The symbol 0 is chosen randomly with probability $p_0$ and the symbol 1 is chosen with probability $p_1 = 1 - p_0$. Notice that the information content (Shannon entropy) is maximized at $p_0 = p_1 = \frac{1}{2}$, the point at which the surprise as to which symbol is chosen is maximum.

one bit.

It may seem strange to talk about fractions of a bit of information. What does that mean? To prevent confusion we have to remember that the word bit has two meanings (see Box 1.1). First it can mean a physical carrier of information that can be in two possible states (e.g. a switch that is on or off). The number of these physical objects is always an integer. The second meaning is an abstract measure of your level of surprise in reading a message. The message is carried by single physical objects storing a 0 or 1. If each object is almost always in state 0 when you read it, then you are not very surprised when the next one drawn from the same probability distribution is also 0. Thus the Shannon information (entropy) carried by one physical instantiation of a bit can be much less than one abstract mathematical bit of information.

Suppose now that Alice sends Bob two symbols each drawn from the same probability distribution. If Eq. (1.9) is correct, then it should predict that the total information transmitted is twice as large as when only a single symbol is sent. We can think of this two-symbol message as being drawn from an 'alphabet' of $M = 4$ possible messages chosen from a probability distribution $R$ shown in Table 1.2.

Here we have used the fact that if two events are statistically independent, their joint probability is the product of their individual probabilities[5]. If Alice

[5]This factoring of a joint probability distribution into two separate probability distri-

| Ordinal number of message | Message | Probability |
|---|---|---|
| 0 | 00 | $r_0 = p_0 p_0$ |
| 1 | 01 | $r_1 = p_0 p_1$ |
| 2 | 10 | $r_2 = p_1 p_0$ |
| 3 | 11 | $r_3 = p_1 p_1$ |

Table 1.2: Enumeration of the four possible messages of length two and their corresponding probabilities.

is using a biased coin toss to select the symbols for the two messages, the second coin toss is unaffected by the outcome of the first. Plugging this into Eq. (1.9) and using the probability distribution $R$ yields

$$\begin{aligned}
H_4 &= -\sum_{j=0}^{3} r_j \log_2(r_j) && (1.11)\\
&= -\left\{p_0p_0\log_2(p_0p_0) + 2p_0p_1\log_2(p_0p_1) + p_1p_1\log_2(p_1p_1)\right\}\\
&= -\left\{2p_0p_0\log_2(p_0) + 2p_0p_1[\log_2(p_0) + \log_2(p_1)] + 2p_1p_1\log_2(p_1)\right\}\\
&= -2\left\{(p_0+p_1)p_0\log_2(p_0) + (p_0+p_1)p_1\log_2(p_1)\right\}\\
&= -2\left\{p_0\log_2(p_0) + p_1\log_2(p_1)\right\}\\
&= 2h(p_0,p_1) = 2H_2. && (1.12)
\end{aligned}$$

Shannon's expression in Eq. (1.9) has precisely the desired property that the information content is linear in the number of messages sent. With a little more work one can show that this expression is the *unique* correct extension of Eq. (1.1) to the case where the messages in the alphabet do not all have the same probability.

butions is essentially the definition of statistical independence: $P(A,B) = P_1(A)P_2(B)$. See App. A

**Box 1.4. Shannon Entropy as the Average Information** Because the Shannon entropy is a function of the entire probability distribution, it makes more sense to view it as the average information content of messages drawn from the probability distribution, rather than the information content of any single message drawn from the distribution. In this picture we can interpret Eq. (1.9) as saying that the information content of the $j$th message is $(-\log_2 p_j)$ bits. The rarer a particular message is, the more suprised you are when you receive it. On the other hand you don't receive it very often so it doesn't contribute much to the average information.

**Box 1.5. Derivation of Shannon's Formula** To derive Shannon's expression we can use the following simple construction to turn a problem of $N$ messages numbered $1, \ldots, N$, all with equal probability into a problem with $g$ messages of varying probabilities. Suppose we have a set of $g$ groups $\{G_1, G_2, \ldots, G_g\}$ (boxes if you like). We take our set of $N$ messages all of equal probability and we assign $n_1$ of them to group $G_1$, $n_2$ of them to group $G_2$, and so on. Clearly $\sum_{j=1}^{g} n_j = N$. Since all the messages are equally likely, each one carries $H = \log_2 N$ bits of information. Equivalently telling the receiver the ordinal number of the message gives the receiver $H$ bits of information. Suppose however instead of telling the receiver the ordinal number of the message, the sender only tells receiver which group the message comes from. We don't yet know how to calculate how much information this transmits, so let's define the information that the message comes from group $G_j$ to be $H_j$. We will shortly see how to compute this.
What is the probability $P_j$ that the message is from group $G_j$? If the messages are selected uniformly at random then clearly $P_j$ is just equal to the fraction of the messages found in group $G_j$ $P_j = \frac{n_j}{N}$. Clearly these messages do not (necessarily) have equal probabilities. Thus we should compute the average information when relaying to the receiver which group the message came from

$$H_G = \sum_{j=1}^{g} P_j H_j. \tag{1.13}$$

The following clever argument allows us to compute $H_j$. Suppose that having told the receiver that the message came from group $G_j$, the send then tells the receiver which of the $n_j$ messages it was. Since they are all equally likely, this news represents $\log_2 n_j$ bits of information. Since the receiver now knows which of the $N$ messages was sent and thus possesses a total of $\log_2 N$ bits of information. From this we deduce that

$$H_j + \log_2 n_j = \log_2 N, \tag{1.14}$$
$$H_j = -\log_2 \frac{n_j}{N}, \tag{1.15}$$
$$H_j = -\log_2 P_j, \tag{1.16}$$
$$H_G = -\sum_{j=1}^{g} P_j \log_2 P_j, \tag{1.17}$$

which is Shannon's formula as stated in Eq. (1.9).

**Exercise 1.1.** Suppose that Alice sends to Bob one message drawn from a code book of $M_1$ messages and having probability distribution $P$, and one drawn from a different code book of $M_2$ messages and having probability distribution $Q$. Show that the average information content of these two messages is simply the sum of the average information content of the individual messages.

$$H = H_P + H_Q$$

This is a generalization of the result in Eq. (1.12).

**Exercise 1.2.** Suppose that Alice sends to Bob $N = 1000$ messages drawn from the following probability distribution

$$P = \{0.93, 0.01, 0.01, 0.01, 0.01, 0.01, 0.01, 0.01\}. \tag{1.18}$$

a) What is the average information content (in bits) of each message?

b) Suppose Alice finds an optimal encoding that allows her to bundle all $1,000$ messages into a single binary code word. How long is that code word? [Strictly speaking, a perfectly optimal encoding that has arbitrarily low failure probability for decoding requires $N$ to be asymptotically large (i.e., $N \to \infty$). We here assume $N = 1000$ is large enough that the decoder almost always works.]

## 1.2 Error Correction and Data Compression

### 1.2.1 Error correction

Error correction is another example of the numerous insights that can be gained from Shannon's work. If the communication channel is noisy (e.g. some bits are randomly flipped from 0 to 1 or vice versa when the message is sent), then there exist useful codes which are *longer* and *less* random than the original message. These redundant encodings are useful because the original message can still be recovered (with high probability) even though the noise adds unwanted randomness. A necessary (but not sufficient) condition for the receiver to be able to decode the message is that the encoded message be long enough to have enough extra information capacity to hold both the original uncorrupted message and the information on what particular errors occurred during transmission. If this is not satisfied no code, no matter how clever, can work. On the other hand this condition is certainly not sufficient because the code has to be very cleverly designed so that the receiver can learn which bits are erroneous without knowing in advance anything about the message other than what redundant encoding was used. These same concepts will be important when we study quantum communication and quantum error correction and so we will delay our study them until then.

Without understanding how to construct (classical) error correction codes, we can however derive a bound on how much longer (and less random) the encoded message has to be if the receiver is going to have a good chance to receive it over a noisy channel and correctly decode it. This so-called Hashing bound is described in Ex. 1.3.

**Exercise 1.3. Hashing Bound for Error Correction of Noisy Channels** Alice has a message $M$ of random bits of length $L_M$ she wishes to send Bob. She encodes this into a *longer* string $M_{\text{encoded}}$ of of length $N$ bits that is *less* random and sends it to Bob through a noisy channel. The channel has a small probability $\epsilon$ of any given bit being erroneously flipped during transmission. We can think of the action of the error channel as increasing the randomness of the bit string in the following way: Alice transmits the encoded message $M_{\text{encoded}}$ of $N$ bits but before Bob receives it, the noise demon creates a random error string of the form $C = 0, 0, 1, 0, 0, 0, 0, 0, 0, 1, \ldots$ of length $N$ in which each bit is chosen randomly to be 1 with probability $\epsilon$ and 0 with probability $1 - \epsilon$. The noise demon then transforms Alice's transmitted bit string into a corrupted bit string $M'$ via

$$M' = M_{\text{encoded}} \oplus C \tag{1.19}$$

where $\oplus$ indicates bitwise addition modulo 2 (i.e., $0 \oplus 0 = 0, 0 \oplus 1 = 1, 1 \oplus 0 = 1, 1 \oplus 1 = 0$).

1. What is the mean $\bar{N}_{\text{e}}$ and standard deviation $\sigma_{\text{e}}$ of the number of errors? Hint: Think of a random walk defined by the bit string $C$ in which the walker steps 0 distance to the right if the bit is correct and 1 step to the right if the bit is erroneous. The final position of the walker is the total number of errors. Note that in the limit where $N$ is so large that the average number of errors is also large, you may use the central limit theorem to argue that the number of errors is (to a good approximation) Gaussian distributed around $\bar{N}_{\text{e}}$ with width $\sigma_{\text{e}} \ll \bar{N}_{\text{e}}$.

2. What is the total Shannon entropy $S_{\text{e}}$ in the probability distribution of the errors (i.e., the entropy in the error string $C$)?

3. If Alice cleverly encodes her message within the string of $N$ bits, Bob should be able to decode it to learn both the string of errors and the bit string of the original intended message $M$ of length $L_M$. Since Alice sends $N$ bits and Bob learns $S_{\text{e}}$ bits about the noise. What is the maximum number of bits $L_M$ that can be in Alice's intended message if Bob is to be able to decode it? This is called the Hashing Bound (valid for asymptotically large $N$). Efficient error correction codes can approach but never exceed this bound.

### 1.2.2 Data Compression

We have seen that randomness is a key feature of information transmission. Without randomness in Alice's choice of messages to send, Bob will not be surprised by what he receives and thus will learn no new information. From Shannon's deep insight many important results follow. For example, the concept of data compression is based on the idea that if the string of symbols in a long message is not completely random, then there exists a *code* (a new set of symbols) that is shorter but more random than the original, that can be used to store or transmit the same information more compactly. Consider for example, the Morse code which was invented for telegraphy by Samuel F.B. Morse (Yale College class of 1810). This is a binary code consisting of 'dots,' 'dashes' with an additional symbol (blank space) separating code words for English letters. The code is transmitted by pulses of current in a wire, with 'dot' corresponding to a brief pulse and 'dash' corresponding to a longer pulse (about 3X longer). The most commonly occurring letter in English text, 'e,' is given the shortest code word, 'dot.' The next most commonly occurring letter, 't,' is given the next shortest code word, 'dash.' A rarely occurring letter like 'z,' is given a longer code word, 'dash,dash,dot,dot.' You could imagine an even more efficient encoding that uses small words to represent commonly used phrases such as commonly done in modern text messaging when LOL is used for 'laugh out loud.'

To better understand classical data compression, let us consider a specific example. Suppose that Alice sends Bob a string of $M \gg 1$ bits each drawn from the probability distribution $p_0 = 1 - \epsilon, p_1 = \epsilon$, where $\epsilon \ll 1$. Further suppose that even though $\epsilon \ll 1$, $M$ is so large that $\epsilon M \gg 1$. Alice's message consists mostly of 0's with occasional 1's sprinkled randomly throughout. However the message is so long that the average total number of 1's, $\epsilon M \gg 1$. Typically there are many consecutive 0's between each bit with value 1, so the message is not very random and the information content is low. The Shannon entropy is in fact

$$H_M = -M\left[(1-\epsilon)\log_2(1-\epsilon) + \epsilon\log_2\epsilon.\right]. \tag{1.20}$$

Rather than Alice transmitting the entire bit string of length $M$, she could save time and instead transmit a *compressed encoding* of the message as a smaller bit string of length $N$. In order for the encoded bit string to contain the same amount of information as the original bit string, it has to be more random than the original bit string. Since the information in the original message is $H_M$, the minimum possible bit string length that a data compression algorithm can theoretically achieve would be $N = H_M$ or equivalently a maximum compression factor $r = M/H_M$. Practical compression codes can approach this limit but never exceed it.

**Exercise 1.4.** A random source outputs 0 with probability $p_0 = 0.37$ and 1 with probability $p_1 = 0.63$. This source is used to construct a string of $M = 2^{20}$ bits (i.e., 1 megabit = $(1024)^2$ bits) which is stored on a hard drive. What is the minimum file size (in bits) that a data compression program could achieve in this case?

## Optional Reading on Possible Encodings

One possible (not necessarily optimal) encoding could be such that each bit string is a binary number indicating the number of consecutive 0's in the original message before the next 1 occurs (in the bit string). Let us try to estimate how much this code compresses the data.

Let $P(\ell)$ be the probability that the next bit with value 1 (in the original message) is separated from the previous occurrence of 1 by $\ell$ zeros. We then have that

$$P(\ell) = (1-\epsilon)^{\ell}\epsilon. \tag{1.21}$$

The first term in the RHS above represents the probability that all of the next $\ell$ bits are zero and the last term represents the probability that the $\ell+1$th bit is 1. Using the properties of geometric series, we can readily prove that (taking $M$ to infinity)

$$\sum_{\ell=0}^{\infty} P(\ell) = 1. \tag{1.22}$$

Let the number of consecutive 0's after a particular 1 be $L$. An $N$-bit binary number can represent any integer from $L = 0$ to $L = 2^N - 1$. Hence our code will succeed provided that $L < 2^N$. The probability of failure is (for each 1 in the message)

$$\begin{aligned} P_{\text{fail}} &= \sum_{\ell=2^N}^{\infty} \epsilon(1-\epsilon)^{\ell} \\ &= (1-\epsilon)^{2^N} \sum_{\ell'=0}^{\infty} \epsilon(1-\epsilon)^{\ell'} \\ &= (1-\epsilon)^{2^N}. \end{aligned} \tag{1.23}$$

This is the failure probability encoding the number of 0's after a particular one of the 1's, in the limit of an extremely long message (since we have taken the upper limits on the summations to be infinity). Because of the double exponential, $N$ does not have to be very large in order to make $P_{\text{fail}}$ extremely small (even if $\epsilon$ is small). It is convenient to approximate the expression for $P_{\text{fail}}$ by taking its (natural) logarithm and stopping its Taylor series expansion at first order in $\epsilon$

$$\ln P_{\text{fail}} = 2^N \ln(1-\epsilon) \approx 2^N(-\epsilon). \tag{1.24}$$

Thus for small $\epsilon$ we have to a good approximation

$$P_{\text{fail}} \approx e^{-\epsilon 2^N}. \tag{1.25}$$

The average number of times we will have to use this encoding in a message string of length $M$ is $\epsilon M \gg 1$. Everyone one of these has to succeed or the message will be garbled. Hence the overall success probability for encoding the entire message is (for the average case)

$$P_{\text{total success}} = (1-P_{\text{fail}})^{\epsilon M} \approx 1 - \epsilon M P_{\text{fail}}, \tag{1.26}$$

where in the last approximate equality we have assumed that $\epsilon M P_{\text{fail}}$ is small. The overall failure probability in this limit is then

$$P_{\text{failure overall}} = 1 - P_{\text{total success}} \approx \epsilon M P_{\text{fail}} \approx \epsilon M e^{-\epsilon 2^N}. \tag{1.27}$$

The length of the original unencoded message is $M$ bits. In the encoded message we are simply sending (on average) $\epsilon M$ bit strings of length $N$. Hence the encoded message has on average a length of $D = \epsilon N M$. Let us compare this to the Shannon entropy of the original message

$$\begin{aligned} H &= M H_2(1-\epsilon, \epsilon) = -M\big[(1-\epsilon)\log_2(1-\epsilon) + \epsilon \log_2 \epsilon\big] \\ &\approx \epsilon\big[1/\ln(2) - \log_2 \epsilon\big] M \ll M, \end{aligned} \tag{1.28}$$

where in the last step we have used $\log_2(x) = \ln(x)/\ln(2) \approx 1.4427\ln(x)$. Naturally, for small $\epsilon$ the information content of the message is much smaller than the length of the message. In principle we should be able to compress the message of length $M$ (nearly) down to a length $D$ approaching $H$, achieving a compression factor approaching

$$C = \frac{M}{H} = \frac{-1}{\big[(1-\epsilon)\log_2(1-\epsilon) + \epsilon\log_2\epsilon\big]} \approx \frac{1}{\epsilon\big[1/\ln(2) - \log_2\epsilon\big]}. \tag{1.29}$$

Let $D = rH$, where $r$ measures how close the code comes to the theoretically maximum compression factor. We know that for any reliable code we need $r > 1$ because we cannot compress a message to a length that is smaller than the Shannon information content of the message. We have (for small $\epsilon$)

$$r \approx \frac{\epsilon NM}{\epsilon M[1/\ln(2) - \log_2\epsilon]} = \frac{N}{1/\ln(2) - \log_2\epsilon}. \tag{1.30}$$

The smaller is $r$ the less reliable is the encoding. However for $r$ slightly greater than unity, the failure probability can be very small for an efficient code.

As a specific example for the code being considered here, suppose that $\epsilon = 2^{-7} \approx 0.78 \times 10^{-2}$. In this case the theoretical maximum compression factor is (using the exact and approximate formulae in Eq. (1.29))

$$C = 15.1712 \approx 15.161. \tag{1.31}$$

Let us take the overall raw (unencoded) message length to $M = 2^{14} = 16192$. Then $\epsilon M = 2^7 = 128 \gg 1$ as required by our assumptions. If we choose $N = 11$ then the overall failure probability from Eq. (1.27) is

$$P_{\text{failure overall}} \approx 2^7 e^{-2^{N-7}} \approx 1.44 \times 10^{-5}, \tag{1.32}$$

and we have for the $r$ factor

$$r \approx 1.3029, \tag{1.33}$$

which is not far above the theoretical bound $r = 1$. If we choose to increase $N$ from 11 to $N = 12$ we obtain a slightly less efficient code $r \approx 1.42135$, but the overall failure probability drops dramatically to

$$P_{\text{failure overall}} \approx 2^7 e^{-2^{N-7}} \approx 1.62 \times 10^{-12}. \tag{1.34}$$

This shows us the remarkable power of the double exponential suppression of the error rate with increasing $N$.

We could slighlty improve the efficiency of the code by not using the truncating the leading zeros in our length $N$ bit strings that define the number of consecutive zeros (see Ex. 1.5). However, if the bit strings are of variable length, one must be able to transmit an additional character to mark the beginning and end of each string. This will come at small but non-zero cost in efficiency. In any case, the overall improvement will be modest since the simple code described here already achieved an $r$ close to unity (at least for the parameters considered here).

The code described above has a serious problem if it does fail. If any one of the bit strings representing the distance to the next occurence of 1 should fail. Then we will no longer be able to find the positions of any of the subsequent 1's. A better idea might be to simply encode the message by listing the positions of each of the 1's (rather than listing their separations). Suppose the message length is $M = 2^N$. If we take position of the first bit in the message to be defined to be position $x = 0$ the range of possible positions for the 1's will be integers in the range $[0, 2^N - 1]$, we can represent them with bit strings of (fixed) length $\log_2 M = N$ without any chance of 'overflow' errors. The average number of 1's will be $\epsilon M$. Hence we can encode the message in a (variable length) bit string whose average length is

$$L = \epsilon M \log_2 M, \tag{1.35}$$

yield a compression factor

$$r = \frac{M}{L} = \frac{1}{\epsilon \log_2 M}. \tag{1.36}$$

Thus for example if $M = 2^{20} \sim 10^6$, we have

$$r = \frac{1}{20\epsilon}, \tag{1.37}$$

which exceeds unity for $\epsilon < 0.05$ grows ever larger as $\epsilon$ shrinks. If $\epsilon \log_2 M > 1$, we have $r < 1$ and this code cannot compress the data. Clearly we do not want to choose $M$ too large because the compression factor (slowly) decreases as $\log_2 M$ increases.

So far we have looked at some specific methods of error correction encoding that may or may not be optimal. It is interesting to ask the question: What is the theoretical limit

**Exercise 1.5.** NEED HELP FINDING THE ERROR IN THIS ARGUMENT. ANY VOLUNTEERS? Compute the efficiency factor $r$ for the code described above in which the length of the strings of consecutive 0's is encoded into variable length binary numbers–that is the leading 0's are dropped from the encoding described above. Ignore the cost of having to transmit an additional symbol (or blank space) to demarcate the beginning and end of these variable length strings. Does this code have any chance of failing? Does it achieve the theoretical $r = 1$?

Hint: Let $Q_N$ be the probability that the number of consecutive 0's is such that $N$ bits are required to represent that number. The mean length is given by

$$\bar{N} = 1 + \sum_{N=1}^{\infty} Q_N N,$$

where the first term accounts for the space that needs to be added between the variable-length binary strings.
Show that

$$\begin{aligned} Q_1 &= P(0) + P(1) \\ Q_2 &= P(2) + P(3) \\ Q_3 &= P(4) + P(5) + P(6) + P(7) \\ Q_N &= \sum_{\ell=2^{(N-1)}}^{2^N-1} P(\ell), \end{aligned} \tag{1.38}$$

where $P(\ell)$ is defined in Eq. (1.21), and the last equation above is valid for any $N \geq 2$. Using the properties of geometric series, find an explicit expression for $Q_N$.

Numerical evaluation for the case $\epsilon = 2^{-7}$ yields $\bar{N} = 7.67724$. The achieved compression factor is $C' = 1/(\epsilon\bar{N}) = 16.6726$ exceeds the theoretical limit $C = 15.1712$ producing an $r$ factor less than unity: $r = 0.909945$. This violation means there is something wrong with the above argument! What has gone wrong??

We have seen in our survey of classical information theory that surprise requires randomness and unpredictability. It turns out that randomness plays central and fundamental role in quantum mechanics as we shall shortly see.

In fact the quantum theory is a theory of randomness and probabilities that is a distinct alternative to the ordinary theory of classical probability and statistics outlined in Appendix A. We will soon see that the predictions of quantum theory are strange and wondrous and seem to make no sense until you get used to them. The only thing they have going for them is that they agree perfectly with experiment!

## 1.3 What is a computation?

We are so used to the apparent magic of modern technology that we may not stop think about what a computation is. For our purposes, a (classical) computer is simply a black box that has an input register into which one can load $N$ bits of information and output register of $M$ bits. The black box fills the output register bits based on some specified function of the input register bits. You could even think of the black box as simply an editor that edits the contents of the input register and places the result in the output register. It is hard to imagine that this seemingly simple functionality is what gives us video calling over the internet, computer games, the ability to predict rocket trajectories, numerically solve differential equations, and myriad other applications that we take for granted.

Computers are programmed in order to execute binary-valued functions

$$f : \{0, 1\}^N \to \{0, 1\}^M. \tag{1.39}$$

How many different such functions are there? That is, given an input of $N$ bits and output of $M$ bits, how many distinct programs are there? Let us begin by enumerating the possibilities when $N = M = 1$. How many distinct functions are there that map the domain $\{0, 1\}$ to the range $\{0, 1\}$? Table 1.3 enumerates the four possibilities.

For each of the 2 input values there are 2 possible choices of output value, so we have a total of four possible functions. Let us now consider the slightly more general case of $N$ input bits, still only $M = 1$ output bits. The number of distinct possible inputs is $S = 2^N$. For each of those inputs we have to choose a value for the output. Let us assign an ordinal number $j \in [0, S-1]$ to each input that corresponds to the binary number represented by the input. Thus the 0th input is $\{0, 0, 0, \ldots, 0\}$, and the $(S-1)$th input is $\{1, 1, 1, \ldots, 1\}$. Hence the function evaluated by the program is defined by a string of $S = 2^N$ bits: $\{b_0, b_1, b_2, \ldots, b_{S-1}\}$, where $b_j$ is the bit value output

| Function | Input | Output | Global Property |
|---|---|---|---|
| IDENTITY | 0<br>1 | 0<br>1 | Balanced |
| NOT | 0<br>1 | 1<br>0 | Balanced |
| ZERO (ERASE) | 0<br>1 | 0<br>0 | Constant |
| ONE (ERASE-NOT) | 0<br>1 | 1<br>1 | Constant |

Table 1.3: Enumeration of the four possible functions of one input bit and one output bit. The ERASE function sets all the output states to 0. We can also think of this as the RESET function used to initialize a register to 0. The ERASE-NOT function applies ERASE and then NOT, thereby setting all the output states to 1. Balanced functions output 0 and 1 equally often. Constant functions always output the same value. For the special case of one input bit and one output bit, every function is either constant or balanced, but in the general of $n$ input bits and one output bit, most functions are neither balanced nor constant. (See Ex. 5.3).

by the the program for the $j$th possible input. Each bit in such a list has two possible values so the total number of possible programs is

$$Z(N, M=1) = 2^S = 2^{2^N}. \tag{1.40}$$

To understand all this, the reader may find it useful to study the specific case shown in Table 1.4 which lists the data associated with all possible functions mapping two bits to one bit.

We see from the above that the number of distinct functions is a double exponential which is a very rapidly growing function of the input register size, $N$. For example, for $N = 10$, $Z(10, 1) \approx 1.8 \times 10^{308}$. The situation is even more dramatic for output register size $M > 1$. In this case our program is defined by $M$ different output strings, each of length $S = 2^N$. Thus the total number of binary digits defining the program is $M2^N$ and the total number of possible programs is

$$Z(N, M) = 2^{M\left(2^N\right)} = \left(2^M\right)^{2^N}. \tag{1.41}$$

This is consistent with our starting analysis above that $Z(1, 1) = 4$. Note that

$$Z(10, 10) \approx 3.5 \times 10^{3082} \tag{1.42}$$

| $\vec{x}$ | $f_0$ | $f_1$ | $f_2$ | $f_3$ | $f_4$ | $f_5$ | $f_6$ | $f_7$ | $f_8$ | $f_9$ | $f_{10}$ | $f_{11}$ | $f_{12}$ | $f_{13}$ | $f_{14}$ | $f_{15}$ |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 00 | 0 | 1 | 0 | 1 | 0 | 1 | 0 | 1 | 0 | 1 | 0 | 1 | 0 | 1 | 0 | 1 |
| 01 | 0 | 0 | 1 | 1 | 0 | 0 | 1 | 1 | 0 | 0 | 1 | 1 | 0 | 0 | 1 | 1 |
| 10 | 0 | 0 | 0 | 0 | 1 | 1 | 1 | 1 | 0 | 0 | 0 | 0 | 1 | 1 | 1 | 1 |
| 11 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 |

Table 1.4: List of all the functions mapping 2 bits to 1. The first column lists the four possible arguments of the function. The remaining columns give the corresponding values of the 16 different functions $f_j(\vec{x})$. Notice that the ordinal number $j$ of function $f_j$ determines the four-bit binary string giving the values of $f_j$ for each of its four arguments. We see that only $f_0$ and $f_{15}$ are constant. There are six balanced functions: $f_3, f_5, f_6, f_9, f_{10}, f_{12}$ having equal numbers of 0 and 1 outputs. The remaining eight functions are neither constant nor balanced.

is a seriously big number!

**Exercise 1.6.** Alice has available the set of all possible binary functions that map $N$ bits to $M$ bits. She draws one at random (with equal probability) and communicates her choice to Bob in a message.
(a) What is the smallest number of physical bits that Alice could use to transmit this message? [The input string corresponding to each output string need not be transmitted if we agree in advance that the list of outputs is ordered corresponding to the inputs being in ascending order.]
(b) What is the Shannon information content of this message?
(c) The channel Alice uses has the capacity to transmit 1 gigabit per second. How long would it take Alice to send the message if $N = 30$ and $M = 30$?
(d) How long would it take Alice to transmit all the possible messages once each?

## 1.3.1 Reversible Gates

Notice that the IDENTITY and NOT functions are invertible (reversible). This simply means that given the output $c$ of the function we can uniquely determine what the input $b$ was

$$\begin{aligned} c &= \text{IDENTITY}(b) \implies b = \text{IDENTITY}(c) = \text{IDENTITY}(\text{IDENTITY}(b)) \\ c &= \text{NOT}(b) \implies b = \text{NOT}(c) = \text{NOT}(\text{NOT}(b)) \qquad (1.43) \end{aligned}$$

That is to say, IDENTITY and NOT are invertible because they happen (in this special case) to be their own inverses

$$(\text{IDENTITY})^2 = \text{IDENTITY} \tag{1.44}$$
$$(\text{NOT})^2 = \text{IDENTITY}. \tag{1.45}$$

Obviously, the ZERO (ERASE or RESET) and ONE (ERASE-NOT) functions are *not* their own inverses since they are idempotent

$$(\text{ZERO})^2 = \text{ZERO} \tag{1.46}$$
$$(\text{ONE})^2 = \text{ONE}. \tag{1.47}$$

Indeed, these gates do not have inverses at all. Given the output, we cannot tell what the input was. It turns out that this irreversibility (the fact that the operation cannot be run 'backwards in time') has profound implications on the fundamental thermodynamics of computation. An irreversible computer *must* dissipate energy, just as friction irreversibly brings a hockey puck sliding across the ice to a stop while converting the kinetic energy of the motion of the puck into heat (random thermal motion of the water molecules in the ice). If we were to make a movie of the hockey puck's motion and then ran it backwards, the viewer would immediately be able to tell that the film was running backwards because they would see the puck spontaneously speeding up (and the ice cooling down!) something that is not physically possible (or at least fantastically statistically unlikely). This is what we mean by irreversible in the physics sense. In the mathematical sense, an irreversible gate is not invertible–it does not have an inverse because there is not enough information at the output to deduce the input.

The thermodynamics of computation were explored by Rolf Landauer and Charles Bennett at IBM and Bennett developed a reversible model (gate set) of classical computation in which the only irreversible step was the erasure of information needed to initialize the computer at the beginning of the computation. As illustrated in Fig. 1.4, this irreversibility is related to the fact that RESET of a register with unknown contents (caused say by the person who ran their program before you in the queue) lowers the entropy (Shannon or thermodynamic) of the register. The second law of thermodynamics says that the entropy of the universe tends to increase and lowering the entropy of a system generally requires you to do work (e.g. with a refrigerator). While these ideas were profound contributions to theoretical physics, in practice classical computers are highly energy inefficient and dissipate many orders of magnitude more energy than the theoretical limits discovered by Landauer and Bennett. The 'friction' in their operation is largely associated with electrical resistance in the wires of the silicon transistor chips.

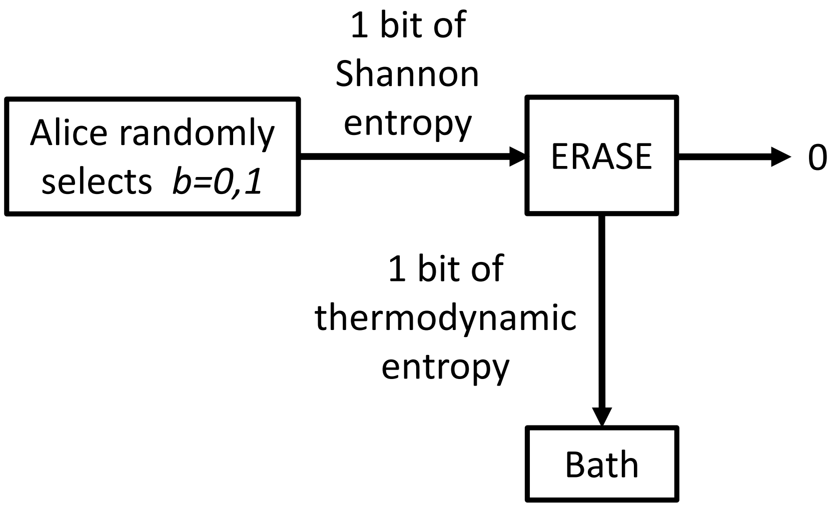
1 bit of Shannon entropy
Alice randomly selects *b=0,1*
ERASE
0
1 bit of thermodynamic entropy
Bath

Figure 1.4: Alice selects a random bit value 0 or 1 and inputs it to an ERASE gate. The output is 0 independent of the input. The input comes from a probability distribution containing one bit of Shannon entropy, but the output has zero entropy. Because the universe conserves information, the missing Shannon entropy must have been irreversibly dumped into bath as thermodynamic entropy.

We will shortly discuss the reversible SWAP gate and Fig. 1.5 shows that RESET can be achieved via such a gate, provided that one has a resource consisting of a supply of bath bits in state 0, and each of those bits is used only a single time. This picture makes clear that the SWAP gate simply moves the entropy (information) from the data bit to the bath bit. If Alice has no further access to the bath after that, then the operation is effectively irreversible[6]'.

As we shall see later, it turns out that all the operations (other than the initialization or RESET) in a quantum computer are required by the laws of

[6]It begs the question however, of how one irreversibly sets the original bath bits to zero in the first place. If one measures the initial value of a bath bit and then applies a NOT operation conditioned on the measurement result being 1, then the initial randomness is converted into information in the hands of the experimentalist, so again information is conserved.

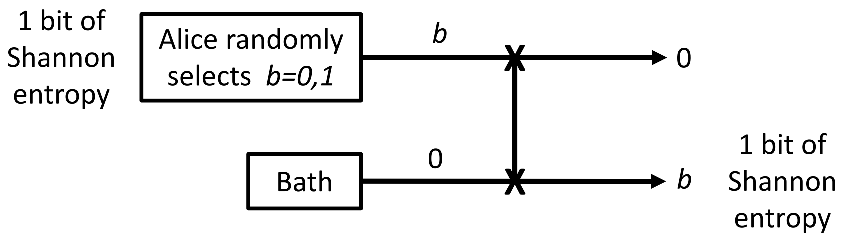

Figure 1.5: Reset can be achieved via the reversible SWAP gate, provided that one has a resource, a supply of bath bits in state 0, and each is used only once. Then it is clear that the entropy has gone from Alice's data bit into the bath.

quantum mechanics to be *reversible.* That is, if a quantum computer dissipates any energy, its program has failed! Thus Bennett's work on reversible classical computation was an important intellectual precursor to quantum computation. It is useful therefore to briefly explore some examples of how to make classical computer operations reversible (in principle). Given that we have seen that RESET destroys information (decreases the Shannon entropy), it is clear that the basic rule for making a reversible computer is simply to never destroy any information. Let us consider as a simple example the AND gate. This gate has two-inputs $x, y$ and a single output which is 0 except if $x = 1$ **and** (hence the name!) $y = 1$. Fig. 1.6 illustrates the circuit.

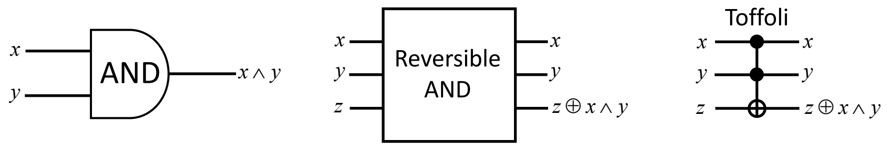

Figure 1.6: Left panel: Standard circuit notation for the AND gate. This is clearly irreversible because there are fewer output lines than input lines and information has been lost. Middle panel: A reversible version of the AND gate that preserves the information by copying the inputs to the outputs and flipping the ancilla bit $z$ if and only if $x = y = 1$. Right Panel: This is also known as a Toffoli or controlled-controlled-NOT (CCNOT) gate, displayed here in the notation we will be using later for quantum gates. This gate applies the NOT operation to the target bit (bottom wire with the open circle symbol) if and only if both control bits (top two wires with the solid circle symbol) are in the 1 state.

These 'circuit' diagrams are schematic representations of the electrical circuits in a computer that produce the corresponding gate. They are read from left to right: the input bits are each represented by a horizontal line ('wire') entering the blackbox from the left, and the output bits are represented by wires exiting on the right. The wedge notation $x \wedge y$ represents Boolean logical AND or equivalently the product of the bit values $xy$. The $\oplus$ notation refers to addition of the bit values modulo 2:

$$\begin{aligned} 0 \oplus 0 &= 0 \\ 0 \oplus 1 &= 1 \\ 1 \oplus 0 &= 1 \\ 1 \oplus 1 &= 0. \end{aligned} \tag{1.48}$$

Note that this is the truth table for the 'exclusive OR' (XOR) gate, which outputs 1 iff one, but not both, of the inputs are 1.

Clearly the traditional AND gate shown in the left panel of Fig. 1.6 is irreversible because there is not enough data in the bit value on the single output line determine the bit values on the two input lines. (See Table 1.5.) The circuit in the middle panel shows a reversible version of the AND gate. It has the same number of output and input lines. This gate is its own inverse since applying it twice yields IDENTITY. This follows directly from algebraic identity

$$b \oplus b = 0 \qquad \forall b. \tag{1.49}$$

Thus the gate is reversible.

Notice that we can reproduce the action of the traditional irreversible AND gate by setting $z = 0$ at the input and discarding $x$ and $y$ on the output, leaving only the single output line $q = x \wedge y$. Also notice that if the input is $z = 1$, then the output is is inverted to NOT(x AND y) = NAND(x,y). The truth table of the NAND gate is given in Table 1.5.

Two other reversible gates are SWAP and CNOT (controlled-NOT) illustrated in Fig. 1.7. As shown in the figure (see also Ex. 1.7), the SWAP gate can be synthesized from three controlled-NOT (CNOT) gates. Just as the name suggests, the SWAP gate interchanges the values of the two bits on its input lines. The CNOT is a simpler version of the Toffoli gate: it peforms a NOT operation on the target bit iff (if and only if) the control bit is 1.

Suppose that we have as a resource a 'bath' of available bits all in state 0. Then the reset operation illustrated in Fig. 1.4 can be performed by simply swapping the data bit into the bath and forgetting about it as illustrated in

| $x$ | $y$ | AND$(x,y)$ | NAND$(x,y)$ |
|---|---|---|---|
| 0 | 0 | 0 | 1 |
| 0 | 1 | 0 | 1 |
| 1 | 0 | 0 | 1 |
| 1 | 1 | 1 | 0 |

Table 1.5: Truth tables for the AND and NAND (NOT[AND]) gates. The NAND gate is illustrated in Fig. 1.8.

Fig. 1.5. This clearly shows that the entropy in the data has been transferred into the bath and thus information is conserved.

See [KLM] Sec. 1.5 for additional discussion of reversible logic.

## 1.4 Universal Gate Sets

We have seen that for a classical computer with an input register of $N$ bits and an output register of $M$ bits, the total number of possible programs that can be run is truly enormous, $Z = 2^{M(2^N)}$. A universal gate set is defined to be any set of gates that can be used to build circuits that execute all $Z$ possible programs (for any size $N, M$). A gate set consisting solely of the Toffoli gate (CC-NOT) is universal for classical computation, provided that one has access to ancilla bits and the ability to initialize these auxiliary bits in 0 or 1 as needed). Since the Toffoli gate is reversible, one can build circuits that execute all $Z$ possible programs and do so reversibly. The reversible circuit will have the same number of input and output lines and their total number may be larger than M+N due to the presence of the ancilla bits.

We have seen that the Toffoli gate is not just universal for reversible classical computations, but can also be used to create irreversible gates if desired (by throwing away the data on the control lines). Another simple gate set that is universal for irreversible computation is {NAND, FANOUT} illustrated in Fig. 1.8. NAND$(x, y)$ = NOT(AND$(x, y)$) and its truth table is shown in Table 1.5. FANOUT splits a single 'wire' into two (or more) wires, or equivalently copies the value of a particular bit into another bit. When we come to quantum gates, we will see that FANOUT is not allowed because of the quantum 'no-cloning theorem.' The quantum Toffoli gate is allowed however because it is reversible and does not violate the no-cloning

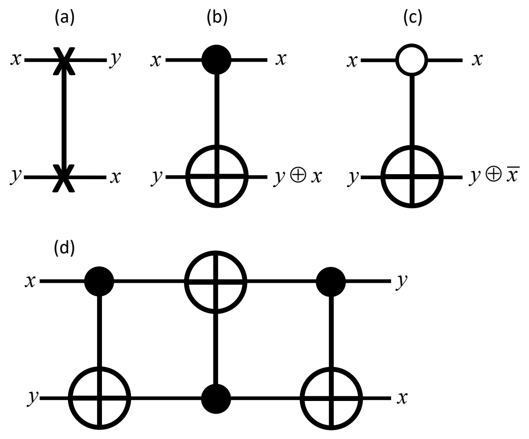

Figure 1.7: (a) Standard circuit notation for the SWAP gate which interchanges the bits on the two wires. (b) The controlled-NOT (CNOT) gate applies the NOT operation to the target bit (denoted by the open circle symbol) if and only if the control bit (denoted by the solid circle symbol) is in the 1 state. The control bit $x$ is unchanged. The target bit $y$ is mapped to $\mathrm{XOR}(x, y) = y \oplus x$. In contrast to the Toffoli gate, the CNOT is a two-bit gate, not a 3-bit gate. (c) $\bar{C}$NOT gate which applies the NOT gate to the target iff the control bit is in the 0 state. $\bar{x}$ denotes $\mathrm{NOT}(x)$. (d) The SWAP gate can by synthesized from three CNOT gates.

rule. Curiously, one- and two-bit gates are inadequate to achieve reversible universal classical computation (hence the need for the three-bit Toffoli gate), however a gate set with only one- and two-qubit gates can be found that *is* universal for quantum computation. Peculiar quantum interference effects permit the Toffoli gate to be synthesized from two-qubit gates, something that is not possible in (reversible) classical computation. More on this later!

**Box 1.6. Computational Complexity** Once we have a universal gate set we can ask the question, how many gates are needed to 'hard wire' a particular one of the $Z(N, M) = 2^{M\left(2^N\right)}$ possible binary functions (programs) enumerated in Eq. (1.41). Said another way, what is the smallest possible circuit depth needed to execute the function correctly for all possible inputs? This is one measure of the computational complexity of executing the desired function[a]. Clearly if the function to be evaluated has a lot 'structure' (as opposed to being 'random') it should be possible to have a very small circuit. For example if the function has $N$ inputs and $N$ outputs and is simply IDENTITY, then we just need an IDENTITY gate on each of the $N$ lines. These can be executed in parallel so the circuit 'depth' is one. On the other hand, if the output value for each possible input is determined by $N$ coin flips (whose results are permanently recorded, not rerun each time the computation is run) then it seems intuitive that the circuit would have to be very complex and deep to deal with each and every possible input in the correct way.

In theoretical computer science, statements about the complexity of solving some class of problems (e.g. the 'traveling salesman' route optimization problem) are typically statements about the asymptotic behavior of the circuit depth as the input size $N$ goes to infinity. There are many classes of problems that contain provably hard (i.e. requiring circuit depth that is superpolynomial in $N$) instances and yet are 'easy' for typical cases. It is often more difficult to prove sharp statements about average-case difficulty than worse-case difficulty.

For a related discussion of complexity of strings of bits in the classical case see:

https://www.wikiwand.com/en/Kolmogorov_complexity

We will see that the complexity of quantum circuits can sometimes be dramatically less than that of the best known classical circuits for certain problems. However there is still much computer science research that needs to be done to fully understand the power of quantum computers and to classify the hardness of different problems on such computers.

[a]Note that in the circuit model, circuit depth and run time are essentially the same thing. The number of lines in a program written in a high-level language need not have anything to do with the run time because of 'DO LOOP' commands. Indeed, the famous 'halting problem' theorem tells us that it is not even possible to write a program that can determine if another program will ever halt.

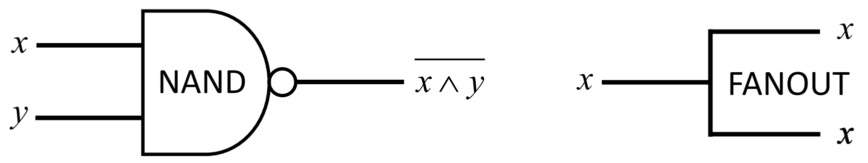

Figure 1.8: Left panel: Standard circuit notation for the NAND (NOT[AND]) gate. Like the AND gate, this is clearly irreversible. Right panel: FANOUT copies the input to two (or more) output lines. These two gates together form a simple but universal gates set for irreversible classical computation. Neither of these gates is allowed in a quantum computer. NAND violates the reversibility requirement and FANOUT violates the quantum no-cloning theorem.

See [KLM] Sec. 1.3 for additional discussion of universal gate sets.

**Exercise 1.7.** Prove that the three CNOT gates shown in Fig. 1.7c are equivalent to the SWAP gate. Do this by considering all four possible input states and explicitly showing in your solution what the states are after each of the three CNOT gates.

**Exercise 1.8.** Use a NAND gate, plus any needed auxiliary bits initialized to 0 or 1, to construct a NOT gate.

**Exercise 1.9.** For a computer with input register of size $N = 2$ and output register of size $M = 2$, there are $Z(2,2) = 4^4 = 256$ possible programs that can be run. One of these is the 'exclusive or' function, XOR, that is true (i.e. has value 1) if and only if exactly 1 of its inputs is 1 (i.e. either $x = 1, y = 0$ or $x = 0, y = 1$). Construct a *reversible* circuit for XOR that is its own inverse

a) using one CNOT gate and 2 wires with the inputs being $x$ and $y$, and the outputs being $x$ and $q = \mathrm{XOR}(x,y) = x \oplus y$, where $\oplus$ is addition mod 2.

b) with the same inputs and outputs as above, but using one Toffoli gate, plus any needed auxiliary bits (internal to the circuit) initialized to 0 or 1. Hint: You will need a total of 3 wires.

c) using two CNOT gates and three wires with the inputs being $x, y, z$ and the outputs being $x, y$ and $q = z \oplus x \oplus y = z \oplus \mathrm{XOR}(x,y)$.

Another gate we will find useful is the controlled SWAP gate (cSWAP) also known as the Fredkin gate, illustrated in Fig. 1.9. cSWAP applies the identity if the control bit is 0 and swaps the two target bits if the control bit is 1. The $\bar{\text{c}}$SWAP does the reverse–performing the SWAP if the control bit is in 0. As discussed in Exercise 1.10, the cSWAP can be constructed from Toffoli gates. As illustrated in Fig. 1.10 and discussed in Exercise 1.11, the cSWAP can be used to construct a router that sends an input bit $b$ down a binary tree to a destination specifed by the bits in an address register.

**Exercise 1.10.** Consider a computer with input register $(x, y, z)$ of size $N = 3$ and output register of size $M = 3$. Construct the controlled SWAP (cSWAP or Fredkin) gate shown in Fig. 1.9 whose output is $(x, y, z)$ if $z = 0$ and whose output is $(y, x, z)$ is $z = 1$. You may utilize one Toffoli gate and two CNOT gates. Hint: With 3 CNOT gates you can create a SWAP. What can you do with 2 CNOT gates and one Toffoli gate?

**Exercise 1.11.** Use three controlled SWAP (cSWAP or Fredkin) gates to construct the binary tree router shown in Fig. 1.10 that sends an input bit $b$ to one of four final destinations based on the values of two address bits, $\{a_1, a_0\}$, and outputs 0 in all the other bits of the output register. Compute how many cSWAP gates would be needed for a general binary tree router with $N$ address bits.

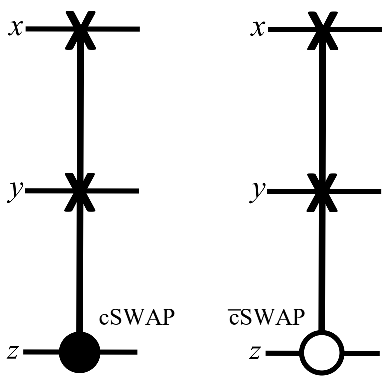

Figure 1.9: Standard circuit notation for the controlled SWAP (cSWAP) or Fredkin gate which swaps the two target bits $x, y$ iff the control bit $z$ is 1. The $\bar{\text{c}}$SWAP gate does the reverse–it swaps the two target bits only iff the control is 0.

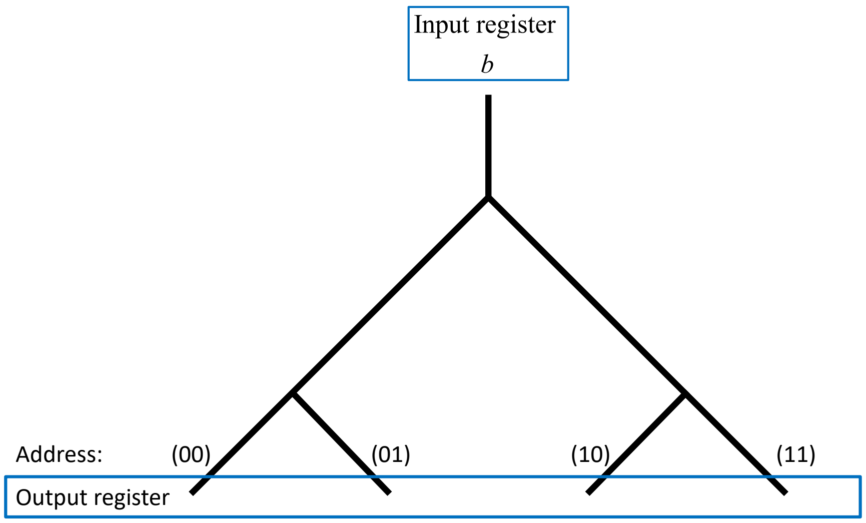

Figure 1.10: A binary tree router that sends input bit $b$ to one of 4 possible elements in the output register based on the values of two address bits $(a_1, a_0)$. All other bits in the output register should be 0.

# Chapter 2

# Quantum Bits: Qubits

As stated previously, information is physical and is stored in the state of a physical system. As illustrated in Fig. 1.1, classical information can be stored in the physical state of an electrical switch and transmitted via the presence or absence of a light from a bulb. As noted previously since a classical bit can take on only two possible values, 0 and 1, there are only two possible physical encodings, 0 and 1 can be represented by the switch off and on respectively, or the other way around. If Alice sends Bob a message via a beam of light and does not tell Bob which encoding she is using, it is an easy matter for Bob to try to decipher the message using both encodings since they differ only by a simple NOT operation that interchanges 0 and 1.

Let us abstract away the details of the physical instantiation of the classical bit and, with a view towards the quantum notation about to be introduced, represent its state by an upward or downward pointing arrow as shown in Fig. 2.1.

As we shall soon see, things are very different for quantum bits. How things are different can only be deduced from the experimental phenomenology. We will present here two versions of the story–the first from a computer science perspective that removes most of the physics language and the second aimed at beginning physicists. The reader is encouraged to learn from both points of view.

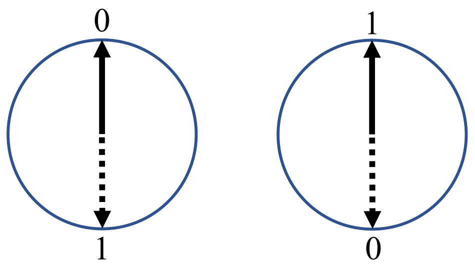

Figure 2.1: Representation of the two possible states of a classical switch (or other physical instantiation of a classical bit) with an up or down arrow. Classically there are only two possible encodings: uparrow is state 0 (left panel) or downarrow is state 0 (right panel). They differ by a NOT operation, or in the language of the arrow, they differ by a rotation of the arrows through 180 degrees.

## 2.1 Quantum Bits for Computer Scientists

A quantum bit ('qubit') is information stored in a quantum system that has two possible physical states, distinguished (say) by the distinct and discretely quantized values of their energy. Quantum bits are like classical bits in that if you measure which energy state they are in, you get a binary result, either 0 or 1. We are perhaps not used to thinking about making measurements of the state of classical bits, yet that is exactly what is done to read out data from a hard drive where the a sensor in the flying head scans over the disk and measures the orientation of the small domains of magnetism whose orientation (up or down, say) represents the state of each bit.

But in addition to this traditional 'digital' (i.e., binary) feature, quantum bits also have 'analog' characteristics because there exists a continuum of possible encodings as illustrated in Fig. 2.2. Associated with this continuous choice of encodings (also known as 'quantization axes' or 'frames') is the ability to choose the measurements we make to be in any of the possible frames. That is, if Bob prepares a qubit in state $0'$ as represented in the right panel of Fig. 2.2, there is a continuum of adjustments he can make to 'orient' his measurement apparatus in order to make its measurements in that same frame. The apparatus contains some controls that determine the unit vector $\hat{n}$ defining the orientation of the frame (in some abstract space, not necessarily our ordinary three-dimensional space).

Something strange happens if Alice prepares state 0 in her frame (left panel of Fig. 2.2) and gives it to Bob to measure in his frame choice. It makes sense that if he chooses $\theta = 0, \varphi = 0$, his frame matches Alice's and he his measurement should yield result 0 since that is what Alice prepared. On the other hand, if Bob chooses $\theta = \pi, \varphi = 0$, then his frame is the same as Alice's except that 0 and 1 are interchanged. Hence his measurement result should always be 1. How do we reconcile the fact that Bob's frame choice can be varied continuously and yet his measurement results will always be discrete? As he changes $\theta$ continuously from 0 to $\pi$, his measurement result has to change from 0 to 1. Since the measurement results are discrete (as for classical bits) it is impossible for them to change continuously. It turns out that the only possible way to reconcile the digital/analog features of quantum bits is through the introduction of *randomness.* The measurement results are always *binary*, but 0 and 1 occur randomly with a probability distribution that varies *continuously* as the measurement 'axis' $\hat{n}$ varies.

Quantum theory is a radical alternative to traditional probability theory and its predictions can be quite different as we will see. Randomness appears in classical probability theory as a representation of our ignorance of certain 'hidden' variables that we would need to know in order to know the outcome of some experiment (e.g., flipping a coin). In the quantum theory, randomness is intrinsic and not due to ignorance. If Alice prepares 0 in her frame and Bob measures in his frame, neither Alice nor Bob have any way of knowing what the outcome of the measurement will be. The result is truly and ineluctably random. This can be a resource that is more powerful than the 'pseudorandom' numbers generated by codes run on classical computers However experiment shows (and quantum theory correctly predicts) that Bob will measure $0'$ and $1'$ with probabilities

$$p_{0'} = \cos^2\left(\frac{\theta}{2}\right) = \frac{1}{2}\left[1 + \cos\theta\right] \tag{2.1}$$

$$p_{1'} = \sin^2\left(\frac{\theta}{2}\right) = \frac{1}{2}\left[1 - \cos\theta\right]. \tag{2.2}$$

These probabilities are plotted in Fig. 2.3. Notice that for $\theta = 0$ there is no randomness in the measurements, and Bob's results always give 0 in agreement with with Alice. For the $\theta = \pi$, there is also no randomness, but Bob's results are always exactly opposite those of Alice. Notice that the randomness is maximized at $\theta = \pi/2$ where Alice's and Bob's quantization axes are perpindicular. At that point Bob's measurement results are a '50;50 coin-toss' and yield the maximum possible information of one classical bit. However this is not information Alice is sending to Bob. She has absolutely

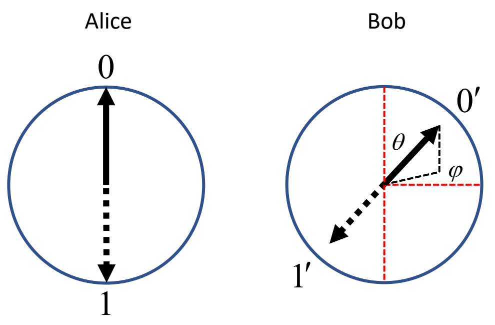

Figure 2.2: There are an infinite number of possible encodings for quantum information. These encodings are in one-to-one correspondence with the points on the surface of a unit sphere. Left panel: Alice chooses the 'standard' encoding. Right panel: Bob chooses an encoding (aka 'quantization axis,' or 'frame') defined in polar coordinates by the polar angle $\theta$ and azimuthal angle $\varphi$ of a chosen point on the unit sphere (also known as the 'Bloch sphere' in honor of Felix Bloch).

no control over Bob's measurement results (i.e., if she prepares 1 instead of 0, it has no affect on Bob's random results if $\theta = \pi/2$.).

There is a second strange aspect of the dual binary/analog nature of quantum bits. Suppose that Bob measures Alice's qubit and obtains the (random) result $0'$. If he measures the *same* qubit again, the result will not be random. It will be identical to the first result. The next time Alice sends Bob a qubit (again prepared in her state 0), Bob might randomly obtain the measurement result $1'$. If he measures that same qubit again, the result will always be $1'$. The randomness has dissappeared! Consistent with the discussion above (in which Bob's original measurement is not random if $\theta = 0$), the only way for the randomness to go away is if the act of measurement has (rather mysteriously) realigned the qubit with Bob's measurement axis (parallel or anti-parallel to it, depending on whether the measurement result is $0'$ or $1'$). *This so-called 'measurement back action' or 'state collapse' is completely invisible to Bob.* Suppose that Bob does not know that his frame is not aligned with Alice's. When Alice sends him a single bit in her state 0, Bob obtains one of his measurement results randomly, but with only one event, he has no way of knowing if this is random. If the result is, say, $1'$, he can verify it by measuring several times and will always get $1'$. As far as he can tell, Alice sent him a $1'$, even though in fact, she sent him a 0.

As we will see later when we study quantum encryption and communication, this invisibility of the measurement back action can be used as a

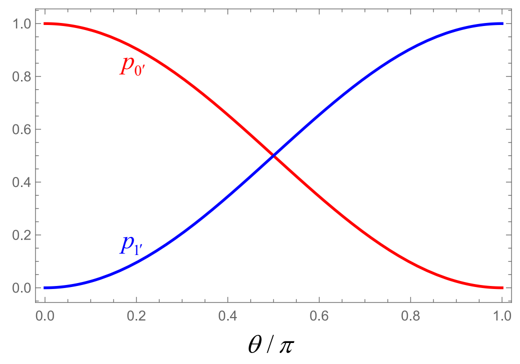

Figure 2.3: Probabilities $p_{0'}, p_{1'}$ that Bob measures results $0, 1$ in his frame when Alice prepares the qubit in state 0 in her frame. $\theta$ is the polar angle between Bob's quantization axis and Alice's. The probabilities are independent of the azimuthal angle $\varphi$.

resource for encrypting communications and for actively detecting the presence of an eavesdropper. Conversely the same effect dramatically complicates quantum error correction in computation (and communication). When you look to see if errors have occurred you do damage to the state!!

**Box 2.1. Holevo Bound** If Alice and Bob agree in advance on the quantization axis to use, Alice can send Bob information by sending him qubits. For simplicity let us say they agree to use the standard computational basis states $|0\rangle$ and $|1\rangle$. If Alice wishes to send Bob the three-bit classical message 011, she can send him three qubits in states $|0\rangle, |1\rangle, |1\rangle$. Because Bob knows the correct quantization axis to use, his measurement results will be $0, 1, 1$ and he will correctly receive the message Alice sent. If Bob uses the wrong quantization axis for his measurement, he will gain less information about Alice's message because randomness will cause his measurement results to (have some non-zero probability to) disagree with what Alice intended.

Alice can (in principle) encode an infinite number of bits in the complex coefficients of a coherent superposition state $\alpha|0\rangle + \beta|1\rangle$, but Bob's measurement can only yield one classical bit of information per qubit. Subsequent measurements on the same qubit yield no new information because of state collapse. This limit of one classical bit of information per quantum bit is called the *Holevo bound* on the transmittable information.

If Alice sends Bob a large collection of qubits, all prepared in the same state, say 0, then Bob can tell something about the alignment of his frame relative to Alice's from the probability distribution of the results. Using his measurements Bob can estimate $\theta$ from

$$p_{0'} - p_{1'} = \cos^2\left[\frac{\theta}{2}\right] - \sin^2\left[\frac{\theta}{2}\right] = \cos[\theta]. \tag{2.3}$$

We will see later that by orienting his frame in a different direction, he can also acquire information about $\cos[\varphi]$, provided Alice supplies him with additional fresh qubits all prepared in the same state.

**Exercise 2.1.** The quantum **Zeno effect** is a vivid demonstration of the existence of measurement back action. Suppose that Alice gives Bob a qubit prepared in state 0 in her frame. Further suppose that Bob's frame is aligned with Alice's frame, so that if he measures the qubit he will also obtain the result 0 with probability 1. Bob can (with high probability) rotate the qubit from state 0 to state 1 purely by making a series of measurements! Suppose Bob makes a series of $N \gg 1$ measurements, gradually rotating his quantization axis between each measurement. Let the first measurement have $\theta_1 = \pi/N$ and the $j$th measurement have $\theta_j = j\theta_1$. (Assume all have $\varphi = 0$.) If all $N$ measurements happen to yield state $0'$ in Bob's (gradually rotating) frame, then after $N$ measurements, Alice will see that Bob's measurements will have turned the qubit from her 0 to her 1 orientation. A lower bound on the success probability is given by the probability that every single one of Bob's measurements is $0'$ (in his gradually rotating frame). Compute this probability as a function of $N$ and find an analytic approximation for it valid for $N \gg 1$. Hint: expand the logarithm of the probability for large $N$ and then exponentiate that.

## 2.2 Quantum Bits for Budding Physicists

A quantum bit ('qubit') is information stored in a quantum system that has two possible physical states. In classical mechanics, the energy of physical systems can vary continuously. Quantum systems however (can[1]) have discrete, or 'quantized' energy levels, as illustrated in Fig. 2.4. These discrete

[1]When an electron is in a low-energy state in an atom, it is 'bound' to the nucleus and has only discrete energy levels. For very high excitation energies the atom becomes ionized and the electron becomes unbound. In that case the allowed energy levels of the electron form a continuum rather than being discrete. For all of the qubits we study, we will be discussing only the low-energy states which are always discretely quantized.

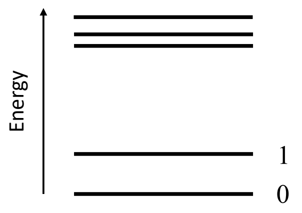

Figure 2.4: Schematic illustration of the quantized energy levels of a naturally occuring (or an artificial) atom used as a quantum bit (qubit). The atom being in the lowest energy state can represent state 0 and the atom being in the first excited state can represent state 1. Electromagnetic radiation can be used to flip the qubit from one state to the other. For natural atoms the required frequency can be in the optical or the microwave. For artificial atoms such as superconducting qubits, the required frequency is in the microwave domain.

physical states can be used to store information. For example, state 0 might denote the atom being in its lowest energy state (the ‘ground’ state), and state 1 might denote the atom being in its first excited state. The quantization of atomic energy levels is related to the fact that the electrons orbiting the nucleus behave like waves. The allowed energies of the electrons can be found by solving the Schrödinger wave equation–a complex task outside the scope of this discussion. Crudely speaking however, one can understand energy level quantization as arising from the fact that in the quantum theory particles act like waves and only electron orbits whose circumference is an integer number of wavelengths are allowed (otherwise the wave destructively interferes with itself). Most quantum systems have many more than two states, but with the right experimental situation, it is possible to focus attention only on the two lowest energy states and ignore the others (to a good approximation).

**Box 2.2. Classical Mechanics vs. Quantum Mechanics** In classical mechanics, the configuration space is the set of all possible configurations of the system. The configuration of the $N$ particles in a system can be described by listing the position vectors $\vec{r}_j = (x_j, y_y, z_j)$ for every particle $j \in \{1, N\}$. These $N$ points in 3-dimensional position space, or equivalently giving a single point in a $3N$-dimensional configuration space, defines the spatial configuration of the system. Because Newton's equation of motion contains two time derivatives (for the acceleration caused by the forces on the particles), the spatial configuration is not enough to predict the future state. You need to also know the momentum vector $\vec{p}_j$ of each particle. The $6N$-dimensional space that defines the positions and momenta of all the particles is called *phase space.*

In quantum mechanics the state ('configuration') of a quantum system is not defined by a point in the usual configuration space. It is defined by a *complex-valued function on that space*–this function is known as the wave function. Because the Schrödinger equation governing the time evolution of the wave function has only one time derivative, we only need to know the wave function $\Psi(\vec{r}, t)$ to predict the future state. [In essence $\Psi$ contains information about the momentum as well as the position.] Max Born gave an interpretation of the wave function: The intensity $\Psi^*(\vec{r}, t)\Psi(\vec{r}, t)$ gives the probability (density) of finding the particle at $\vec{r}$ (or finding the positions of all the particles if $\vec{r}$ is a $3N$-dimensional vector in the case of an $N$-particle system). Because the Schrödinger equation is linear, the sum of two wave solutions is also a solution, leading to the possibility of quantum interferences (since the magnitude squared of the sum of the waves is not equal to the sum of the individual probability densities of each wave).

For simplicity let us consider one spatial dimension and $x \in [0, L]$. We can if we wish, approximate the complex function $\psi(x)$ by specifying its value at $M$ equally spaced points over the interval and approximating the function as piecewise constant. Then the function is defined by a complex vector of (very large) length $M$; that is, the function is (approximately) represented by a vector in an $M$ dimensional complex vector space. See the discussion in R. Shankar, *Principles of Quantum Mechanics.*

The configuration space of a classical bit consists of just two points, labeled 0 and 1. The state (configuration) of a qubit is defined by a complex-valued function on those two points (which in the Dirac notation we call $|0\rangle$ and$|1\rangle$). A function whose argument takes on only two values returns two complex numbers and these two numbers completely specify the function. We can call them $\psi_0$ and $\psi_1$ and think of the state defined by these two values as being represented by a two-component complex vector

$$|\Psi\rangle = \begin{pmatrix} \psi_0 \\ \psi_1 \end{pmatrix} = \psi_0|0\rangle + \psi_1|1\rangle.$$

One of the key features of quantum mechanics is that if one measures the energy of an atom (or more generally any qubit), one *always* obtains a result equal to one of the quantized values allowed for the energy and never anything else. If we are allowed to focus on only the two lowest levels then we always get one of two possible measurement results corresponding to the qubit being in state 0 or state 1. So far this is exactly the same as for classical bits. For example, one can measure the switch setting in the circuit shown in Fig. 1.1 by checking to see if the light bulb is on or off. In principle therefore it is possible to build a classical computer using individual atoms to store bits. This would be the ultimate in miniaturization!

However, because quantum systems behave strangely–particles can act like waves and waves sometimes act like particles, quantum bits are completely different than classical bits in that they have both analog and digital characteristics. These differences will give rise to the surprising powers of quantum systems to store, manipulate and transmit information. The wave-like features of quantum systems means that a qubit can exist in an infinite number of different 'superposition' states intermediate between 0 and 1. Mathematically this goes back to the fact that the Schrödinger wave equation is a linear differential equation. Therefore if it has two (or more) different solutions, any arbitrary linear combination of those solutions also solves the equation. You have seen the superposition principle in action if you have ever watched two waves on the surface of a lake pass through each other and keep going as if nothing had happened. You have also heard it in action when two people are speaking and the vibrations of the air (the sound waves) from each speaker reach your ears.[2]

Fig. 2.5 illustrates the interference effects which occur when two waves are superposed.

For two-state systems (qubits) superposition states are represented mathematically in a notation invented by Paul A.M. Dirac as

$$|\psi\rangle = \alpha|0\rangle + \beta|1\rangle. \tag{2.4}$$

Here $|0\rangle$ represents state 0 of the qubit and $|1\rangle$ represents state 1 of the qubit, and $\alpha$ and $\beta$ are (complex) wave amplitudes also known as 'probability amplitudes.' The 'state' of the qubit has something to do with the kind of

[2]The physics of the sound waves is such that, to an excellent approximation, each one propagates through the air unaffected by the other. This does not mean that your brain won't have trouble following two conversations at once, but that is a separate matter.

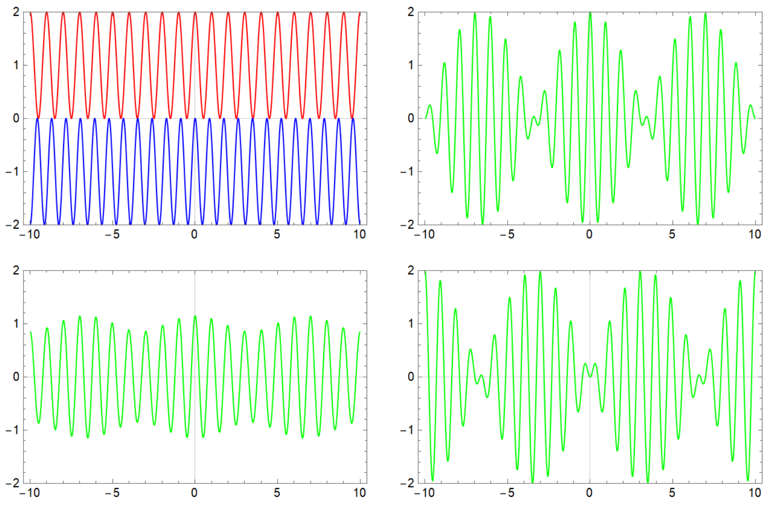

Figure 2.5: Top left panel: Two sine waves, $\Psi_1$ and $\Psi_2$ of unit amplitude (vertically displaced from each other for clarity) having slightly different wavelengths. Top right panel: Superposition of the two waves $\Psi = \alpha\Psi_1 + \beta\Psi_2$, with $\alpha = \beta = +1$. The amplitude is the sum of the amplitudes of the two waves. In some regions the waves cancel each other out (destructively interfere) and in other regions they are in phase and constructively interfere. Bottom left panel: Interference pattern when the wave amplitudes are unequal, $\alpha = +1, \beta = +0.25$. The interference effects are thus smaller. Bottom right panel: Superposition of the two waves $\Psi = \alpha\Psi_1 + \beta\Psi_2$, with $\alpha = +1, \beta = -1$. The net amplitude is the difference of the amplitudes of the two waves. Notice that the regions of high net amplitude are different than in the upper right panel.

waves illustrated above, but for our purposes we do not ever need the details of that. $|0\rangle$ and $|1\rangle$ are abstractions of these wave states and all we have to know about them is that they represent two distinct energy states of the physical object holding the quantum bit of information. As we shall see later, the complex wave amplitudes $\alpha$ and $\beta$ do control interference effects which are analogous to, but an abstraction from, those shown in Fig. 2.5. The analogy for qubits to take away from this is the understanding that quantum superposition states such as

$$|\Psi_+\rangle = \frac{1}{\sqrt{2}}\left[|0\rangle + |1\rangle\right], \tag{2.5}$$

and

$$|\Psi_-\rangle = \frac{1}{\sqrt{2}}\left[|0\rangle - |1\rangle\right], \tag{2.6}$$

are physically distinct, just as the wave patterns corresponding to the different superpositions shown in Fig. 2.5 are distinct.

Readers familiar with the concept of atomic orbitals in chemistry may find it useful to think about the waves associated with the ground (S orbital) and excited ($\mathrm{P}_x$ orbital) states of the hydrogen atom as illustrated schematically in Fig. 2.6. The two LCAOs (linear combination of atomic orbitals) corresponding to $|\Psi_\pm\rangle$ correspond to distinct SP hybridized bonds sticking out to the right and left as illustrated in Fig. 2.6. It is important to note that while these two states of the hydrogen atom could in principle be used to form a qubit, the actual physical qubits in quantum hardware could be completely different. The states $|0\rangle$ and $|1\rangle$ are simply abstract representations of whatever the two lowest energy states of our quantum bit are. The mathematics describing quantum two-level systems is the same for all qubits and the implementation details are not needed for discussion of the theory.

Readers unfamiliar with these quantum concepts, need not worry about them. We will provide the rules of the abstract game as a generalization of the rules for manipulating classical bits learned in computer science.

Qubits are analog devices in the sense that they have a continuum of possible superposition states defined by two complex numbers (here $\alpha$ and $\beta$) and yet they are digital in the sense that if you measure which energy state they are in there are only two possible results for the measurement, $|0\rangle$ and $|1\rangle$. (That is, the measured energy is always found to be either $E_0$ or $E_1$.) There is a peculiar asymmetry here–it takes an infinite number of

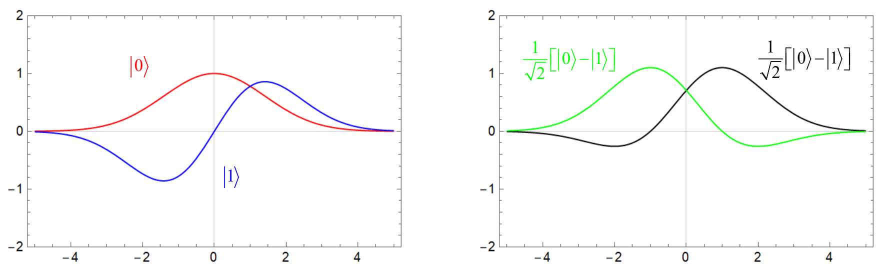

Figure 2.6: Left panel: Schematic illustration of the wave describing the ground state $|0\rangle$ and lowest excited state $|1\rangle$ of the electron in a hydrogen atom. (Sharp-eyed expert readers will notice that these are actually the wave functions for a one-dimensional harmonic oscillator.) Right panel: waves corresponding to two different linear combinations (superpositions) of the two orbitals. One superposition would be appropriate for forming a chemical bond sticking out to the right and the other to the left.

bits to specify the quantum state (i.e. specify the complex numbers $\alpha, \beta$) but a measurement of the state always yields only one of two results. Thus the information gained from the measurement is (at most) one (classical) bit. How can we reconcile these wildly different continuous and discrete properties of qubits? It seems reasonable that if $\alpha = 1$ and $\beta = 0$ so that we have the state

$$|\psi\rangle = |0\rangle, \tag{2.7}$$

then we should always measure the energy to be $E_0$ and never $E_1$. Conversely, if $\alpha = 0$ and $\beta = 1$, so that we have the state

$$|\psi\rangle = |1\rangle, \tag{2.8}$$

then we should always measure the energy to be $E_1$ and never $E_0$. But what happens if we continuously decrease $\alpha$ from 1 to 0 and continuously increase $\beta$ from 0 to 1? At what point does the measurement result change from being $E_0$ to suddenly being $E_1$? Given that the change in the superposition state is continuous, it seems like the measurement results ought to also be continuous. It turns out that the only way we can reconcile the discreteness of the measurement results with the continuity of the superposition states is for the measurement results to be *intrinsically and ineluctably random.* The measurement results are still discrete, but the *probabilities* of obtaining $E_0$ and $E_1$ vary continuously with $\alpha$ and $\beta$. Without randomness there is simply no way to reconcile the continuous behavior and the discrete behavior. This is the source of randomness in the quantum theory.

If one measures whether the qubit has value 0 or 1, the answer is random (provided $\alpha$ and $\beta$ are both non-zero). There is simply no way to predict the outcome of the measurement. As discussed in Box 2.4, this randomness is intrinsic to the quantum theory and is *not* the result of ignorance of the values of any hidden variables that Alice forgot to set or Bob failed to measure. Max Born[3] argued that $\alpha$ and $\beta$ should be thought of as wave *amplitudes* and the measurement probabilities are given by the wave *intensities*. That is, the so-called Born rule states that the measurement yields 0 with probability[4] $p_0 = |\alpha|^2$ and 1 with probability $p_1 = |\beta|^2$.

We know from ordinary probability theory that if $A$ and $B$ are mutually exclusive[5] events with probabilities $p_A, p_B$ respectively, then the probablity of $C = (A \text{ OR } B)$ is

$$p_C = p_A + p_B. \tag{2.9}$$

From this we have the important inequality

$$p_C \geq \max\{p_A, p_B\}. \tag{2.10}$$

Since the events of measuring 0 and 1 exhaust the set of all possible outcomes, we have the important constraint that the two measurement probabilities must add up to unity

$$|\alpha|^2 + |\beta|^2 = 1. \tag{2.11}$$

We say that the state $|\psi\rangle$ must be 'normalized.' To repeat: As $\alpha$ and $\beta$ vary continuously, the probabilities $p_0, p_1$ vary continously but the measurement results are always one of two discrete values, $E_0, E_1$.

Since classical probablities are positive, we know that if there are two mutually exclusive ways an event can occur, the overall probability will be increased. This can be seen in Eq. (2.9). If $\{A, B, C, D\}$ are four possible mutually exclusive outcomes but $\{A, B\}$ are in the category 'good' and

[3]Interesting trivia item: Max Born (who won the Nobel Prize in 1954) was the grandfather of British-Australian singer, Dame Olivia Newton John, who co-starred with John Travolta in the 1978 musical movie, *Grease*.

[4]We are using the following standard notation. If we have a complex number $z = x+iy$, we define complex conjugation as $z^* = x - iy$ and the magnitude (squared) of the number as $|z|^2 = z^*z = x^2 + y^2$.

[5]Mutually exclusive simply means that $A$ and $B$ never both occur on any 'throw of the dice.'

$\{C, D\}$ are in the category 'bad,' then there are two ways to have a good outcome and the probability of such an outcome is given by the sum of the probabilities of all the different ways a good outcome can occur. Consider however what would happen if there were two contributions to the quantum probability *amplitudes* instead of the probabilities

$$|\Psi\rangle = \frac{1}{\sqrt{\Lambda}}\left[(\alpha + \alpha')|0\rangle + (\beta + \beta')|1\rangle\right]. \tag{2.12}$$

We interpret the probability amplitudes as (complex-valued) wave-like amplitudes[6]. Noting that wave amplitudes can interfere constructively or destructively, we see that the probability of a particular outcome might be increased or decreased if there are two contributions to the probability amplitude. We can have

$$|\alpha + \alpha'|^2 = |\alpha|^2 + |\alpha'|^2 + \alpha^*\alpha' + \alpha'^*\alpha \tag{2.13}$$

greater than (constructive interference), equal to, or less than (destructive interference) $|\alpha|^2 + |\alpha'|^2$. Thus quantum systems can violate the inequality in Eq. (2.10). Unlike probabilities, probability amplitudes can be negative or even complex. *Thus it is possible that when there are two ways for something to happen (two 'paths' to the same final state) they cancel each other and the event never occurs!* This changes the game completely and we will see later that such interference effects play a key role in the functioning of quantum algorithms.

It is important to recognize that an overall 'global phase' of the form

$$e^{i\chi}|\psi\rangle = e^{i\chi}\alpha|0\rangle + e^{i\chi}\beta|1\rangle \tag{2.14}$$

(where $\chi$ is an arbitrary real number) can have no measurable effect because the probabilities for different measurement results depend only on the magnitude squared of the state coefficients (probability amplitudes). Hence, without loss of generality we can choose the phase angle $\chi$ to make the coefficient $a = e^{i\chi}\alpha$ in front of $|0\rangle$ real. The complex number $\beta = b + ic$ is

[6]Here $\Lambda$ is a real-valued number called the normalization factor that must be inserted to guarantee that the probabilities add up to unity. Because of this, there is not really a concept of translations in Hilbert space. The summation of the two amplitudes shown here, once normalized actually corresponds to a rotation that keeps the length of the vector fixed. More on this later in the discussion surrounding Eq. (3.83).)

parameterized by two real numbers $c$ and $d$. However the normalization constraint $a^2+b^2+c^2=1$, removes one of these three degrees of freedom. In the end, the irrelevance of the overall phase and the normalization constraint reduce the four real parameters (defining the two complex numbers $\alpha, \beta$) down to two real parameters to specify the state of a single qubit.

The standard parametrization of basis states associated with the quantization axis $\hat{s}$ in terms of two (real) angles $\theta$ and $\varphi$ is the following

$$|+\hat{s}\rangle = \cos\frac{\theta}{2}|0\rangle + e^{i\varphi}\sin\frac{\theta}{2}|1\rangle, \tag{2.15}$$

$$|-\hat{s}\rangle = \sin\frac{\theta}{2}|0\rangle - e^{i\varphi}\cos\frac{\theta}{2}|1\rangle. \tag{2.16}$$

The two angles $\theta$ and $\phi$ can be visualized as the polar and azimuthal angles of a unit vector given in Cartesian coordinates by

$$\hat{s} = (x, y, z) = (\sin\theta\cos\varphi, \sin\theta\sin\varphi, \cos\theta) \tag{2.17}$$

and pointing from the origin to the surface of the unit sphere as illustrated in Fig. 2.7. The unit vector $\hat{s}$ is variously referred to as the qubit 'polarization vector,' or the 'spin' or 'spin vector' of the qubit[7]. We will later describe the meaning of the unit vector on the Bloch sphere in relation to the properties of the corresponding quantum state and will also explain why half angles (which are common in spherical trigonometry) appear in the parametrization of the state in Eqs. (2.16-2.15). We simply note here that the two states of a classical bit are represented by the Bloch sphere vectors corresponding to the north pole $\hat{s} = (0, 0, 1)$ and the south pole $\hat{s} = (0, 0, -1)$. Quantum bits can be in states corresponding to an arbitrary point on the sphere.

[7]Physics students will know that the word 'spin' refers to the fact that certain elementary particles like the electron carry an intrinsic angular momentum which is a vector quantity that can point in any direction, and yet when we measure the projection of that spin vector onto a fixed axis, we always obtain only one of two results (at least for the electron since it has spin $s = 1/2$ and therefore $2s + 1 = 2$ independent states). The spin degree of freedom of an electron is therefore a qubit! Other kinds of qubits do not literally have an angular momentum vector but their superposition states can still be represented mathematically as if they did.

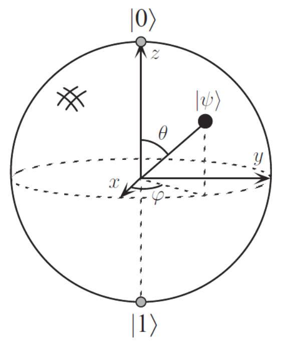

Figure 2.7: Unit vector corresponding to a point on the Bloch sphere with cartesian coordinates $(x, y, z) = (\sin\theta\cos\varphi, \sin\theta\sin\varphi, \cos\theta)$. The orientation of this 3D unit vector can be obtained by starting with the unit vector pointing to the 'north pole' of the sphere and then rotating it in the $xz$ plane by an angle $\theta$ around the $y$ axis and then rotating by an angle $\varphi$ around the $z$ axis. This unit vector corresponds to the parametrization of the qubit state given in Eqs. (2.16-2.15) [Figure Credit: Nielsen and Chuang].

**Box 2.3. A word about notation.** Since $|0\rangle$ and $|1\rangle$ might represent the ground and first excited states of a quantum system, we will sometimes denote them $|g\rangle$ and $|e\rangle$ respectively. Similarly, we may use the orientation on the Bloch sphere to label the same states $|\uparrow\rangle$ and $|\downarrow\rangle$. The state corresponding to $(x, y, z) = (1, 0, 0)$ (or equivalently, $\theta = \pi/2, \varphi = 0$) on the Bloch sphere might be written

$$|\rightarrow\rangle = |+X\rangle = |+\rangle = \frac{1}{\sqrt{2}}[|0\rangle + |1\rangle] = \frac{1}{\sqrt{2}}[|\uparrow\rangle + |\downarrow\rangle], \tag{2.18}$$

It is a weird feature of quantum spins that a coherent superpostion of up and down points sideways!

The state corresponding to the diametricaly opposite point on the Bloch sphere $(x, y, z) = (-1, 0, 0)$ (or equivalently, $\theta = \pi/2, \varphi = \pi$) is

$$|\leftarrow\rangle = |-X\rangle = |-\rangle = \frac{1}{\sqrt{2}}[|0\rangle - |1\rangle] = \frac{1}{\sqrt{2}}[|\uparrow\rangle - |\downarrow\rangle]. \tag{2.19}$$

It is important not to confuse $-|+X\rangle$ with $|-X\rangle$. In the former case the minus sign denotes the quantum amplitude associated with the state pointing in the $+X$ direction on the Bloch sphere. In the latter case, $|-X\rangle$ is a completely different state corresponding to a different point on the Bloch sphere.

Similarly, the states corresponding to the points $(x, y, z) = (0, \pm 1, 0)$ (or equivalently, $\theta = \pi/2, \varphi = \pm\pi/2$) on the Bloch sphere are

$$|\pm Y\rangle = |\pm i\rangle \;=\; \frac{1}{\sqrt{2}}[|0\rangle \pm i|1\rangle] = \frac{1}{\sqrt{2}}[|\uparrow\rangle \pm i|\downarrow\rangle]. \tag{2.20}$$

**Box 2.4. True Randomness** It is very important to understand that the randomness in quantum mechanics does not arise from our lack of knowledge of all the variables in the experimental set up. It is not true that the qubit has a certain value determined by some 'hidden' variable that you do not know about. You can prepare two qubits in exactly the same state and still get different measurement results! In fact, the qubit does not have a 'value' before it is measured. We will see later that assuming that the qubit has a value before it is measured leads to contradiction with experiment. One can actually prove experimentally that there are no hidden variables. This leads to the following clever statement (due to Alexander Korotkov):

'In quantum mechanics you do not see what you get, you get what you see.'

Importantly, if (using an idealized apparatus) you measure a qubit and obtain the random result that it is in state $|1\rangle$ (say), and you then measure the same qubit again, you will get the same value for the second (and all subsequent) measurement results. This means that the first measurement has 'collapsed' the state. If you start with the state $\alpha|0\rangle + \beta|1\rangle$ and the first measurement result happens to be 1, then the state collapses to $|1\rangle$ and stays there for all subsequent measurements. Conversely, if the first measurement happens to yield the value 0, then the state collapses to $|0\rangle$ and stays there. Beginning students find this collapse effect confusing, but don't worry you are in good company, as experts do too!

As stated previously, the above results illustrate a curious asymmetry in quantum information. The state of a qubit requires specification of two real numbers (the Bloch sphere angles, $\theta$ and $\varphi$). In principle this means it takes an infinite number of bits to specify the state. And yet, when we make a measurement, the state collapses to $|0\rangle$ or $|1\rangle$ and we learn only one bit of information. This asymmetry has far-reaching implications as we will see later. In particular it will play an important role in the no-cloning theorem (which tells us that unknown quantum states cannot be copied). This in turn is the basis of quantum cryptography.

One might wonder why one would want to build a computer out of qubits whose measured values seem to be random. The answer (somehow) partly lies in the ability of a collection of qubits to explore a huge number of configurations simultaneously. Thus a collection of 3 qubits can be in a simultaneous superposition of $2^3$ states of the form

$$|\psi\rangle = \alpha_0|000\rangle + \alpha_1|001\rangle + \ldots + \alpha_7|111\rangle. \tag{2.21}$$

Thus if we build a quantum computer with an input register of $N$ qubits, we can give the computer a single giant superposition of all the possible (classical) inputs we could ever give it. Because the Schrödinger equation describing the time evolution of the quantum state of the computer is linear, the computer will carry out its calculation on every possible input simultaneously without difficulty. Thus we have what appears to be exponentially powerful 'quantum parallelism' in which the computer carries out $2^N$ computations at once. Unfortunately, the output register contains a huge superposition of all $2^N$ answers (results of the calculation). When we measure the output register the state randomly collapses to only one of the $2^N$ answers and it seems we lose all the quantum advantage of the parallelism.

It turns out however that all is not necessarily lost. There exist certain classes of problems which have simple answers and one can use the wave-like properties of the superpositions, illustrated in Eq. (2.12), to yield constructive interference which 'focuses' the probability amplitudes in the output register onto the desired answer (see Fig. 2.8). The classic example of this is the Shor algorithm for finding the prime factors of a composite number. The input to the quantum computer is very simple–the single number to be factored. The output is also very simple–one of the factors[8]. The algorithm almost perfectly focuses the output amplitude onto one of the factors so that measurement of the output register gives the correct answer with relatively high probability[9]. Since factoring is a 'one-way problem' that is hard to solve but the solution is easy to check (using a simple classical computation), one simply runs the algorithm a small number of times and checks the answer each time. Within a relatively small number of tries, you will be able to verify that you have found a factor. In this case, where the inputs and outputs are relatively simple, the power of the quantum computer (presumably) comes from its ability to explore an exponentially large number of intermediate states during the course of the calculation.

The Shor algorithm is exponentially faster than the best known classical algorithm for factoring. (Factoring is believed to be exponentially hard classically, but this has not actually been rigorously proven.) Other algorithms offer only polynomial speed up. For example, we will show later that the

[8]This is not quite true. It requires some classical postprocessing of the output to obtain one of the factors.

[9]Actually, the algorithm focuses the output onto many different values, but there exists a fast classical algorithm for finding the desired answer from any one of these outputs using modular arithmetic.

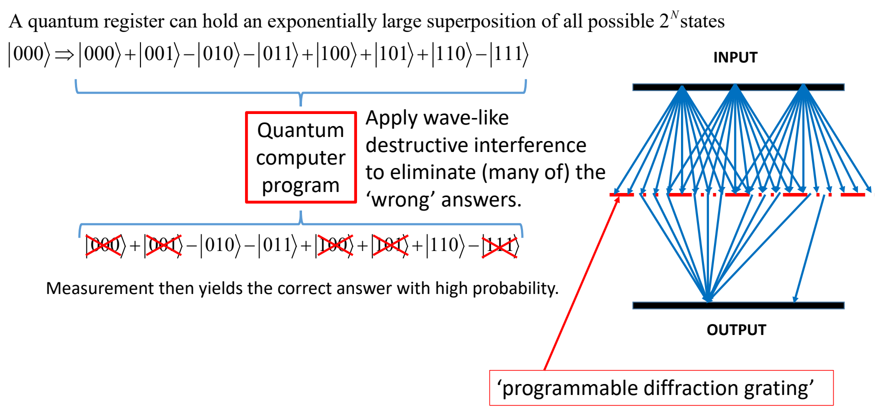
A quantum register can hold an exponentially large superposition of all possible $2^N$ states
$|000\rangle \Rightarrow |000\rangle+|001\rangle-|010\rangle-|011\rangle+|100\rangle+|101\rangle+|110\rangle-|111\rangle$
Quantum computer program
Apply wave-like destructive interference to eliminate (many of) the 'wrong' answers.
$\cancel{|000\rangle}+\cancel{|001\rangle}-|010\rangle-|011\rangle+\cancel{|100\rangle}+\cancel{|101\rangle}+|110\rangle-\cancel{|111\rangle}$
Measurement then yields the correct answer with high probability.
INPUT
OUTPUT
'programmable diffraction grating'

Figure 2.8: Schematic illustration of the action of a quantum algorithm in focusing the wave-like quantum input onto the desired answer in the output register. We can think of the algorithm as a programmable diffraction grating that blocks certain combinations of waves or changes their phase to modify the resulting interference.

Grover search algorithm can find a given entry in an unordered data base of size $N$ in about $\sqrt{N}$ queries to the (quantum) database. Consider for example a telephone directory which has the entries in random rather than alphabetically order. To find your friend's name in this database classcially would require examining on average $N/2$ out of the $N$ total entries. For this problem, the quantum algorithm is not able to focus all the output amplitude onto the desired answer, but rather can partially focus the output probability amplitudes so that they are spread over only $\sqrt{N}$ output states, rather than all $N$. Thus measurement of the output register has probability $p \sim 1/\sqrt{N}$ of collapsing onto the desired state. One thus expects to find the correct answer in about $\sqrt{N}$ rather than $N/2$ attempts as in the classical case. This is not exponential speed up but it is still an interesting (and surprising) quantum advantage.

As mentioned previously, it is important to understand that we still do not yet know the full power of quantum hardware relative to classical hardware. Many quantum algorithms remain to be discovered and the theoretical classification of the hardness (complexity) of different tasks on quantum hardware is not yet complete and remains an important open problem in quantum computer science. Fig. 2.9 gives some illustrative examples.

The source of the power of quantum computers lies partly in the existence of superpositions but also in another subtle concept called entanglement that

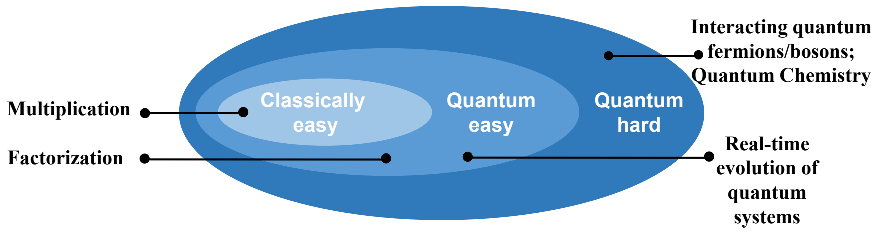
Multiplication
Factorization
Classically easy
Quantum easy
Quantum hard
Interacting quantum fermions/bosons; Quantum Chemistry
Real-time evolution of quantum systems

Figure 2.9: Schematic illustration of quantum and classical problem complexity. Given a quantum state (loaded into the input register of a quantum computer) and the Hamiltonian (energy function) that determines its time evolution via the Schrödinger equation, a quantum computer can efficiently evolve the state forward in time. Thus quantum dynamics is quantum easy. However given the quantum Hamiltonian it may be quantum hard to find the ground state. [Figure courtesy of Shruti Puri.]

gives qubits 'spooky' correlations that can be stronger than is allowed for classical bits. In turns out that to utilize the full power of entanglement requires an additional ingredient which goes by the unfortunate name of 'magic.' Of course it isn't magic, it is physics, but it does seem like magic and Einstein thought that this was sure proof that the quantum theory was wrong. Ironically, today we use this magic as a daily engineering test to make sure our quantum computers really are quantum. We will look into these mysteries in a later section.

**Box 2.5. Can a quantum computer do everything a classical computer can do?** We will see that quantum computers have special powers, but a question one may ask is whether a quantum computer could be used as a classical computer. The answer is yes–the powers of a quantum computer are a superset of the powers of a classical computer. If you load the input register with a standard basis state (not a superposition of such states), quantum Toffoli gates will map such states to other standard basis states without ever creating any quantum superpositions. The action of the Toffoli gates then is equivalent to those of such gates acting on classical bits and we know that this is universal for classical computation.

**Box 2.6. Church-Turing Thesis** In the 1930's Church and Turing made foundational contributions to logic and computer science. Turing invented a (theoretical) prototype computer which is universal–it's capabilities essentially define what is 'computable.' Scott Aaronson summarizes the (physical) Church-Turing Thesis as saying that every physical process can be simulated by a Turing machine to any desired precision, and the Extended Church-Turing Thesis as saying that every physical process can be efficiently simulated by a Turing machine. We now understand that this thesis is *false* for quantum processes. There are quantum processes which are exponentially hard to simulate on classical computers. An interesting question is whether a quantum Turing machine obeys the Extended Church-Turing thesis. It is natural to presume that quantum hardware can simulate any physical quantum process. But is quantum mechanics the ultimate theory that describes all possible physical processes (even those occuring at extreme energies (the Planck scale) where quantum gravity may become important)? Does the universe actually obey some other theory which cannot be efficiently simulated on hardware obeying the rules of ordinary quantum mechanics. I for one, don't know...

# Chapter 3

# Introduction to Hilbert Space

We have seen that there exist a continuum of possible encodings (choice of quantization axis) that can be used to represent and measure the quantum states of a two-level system (qubit). The choice of quantization axis is represented by a three dimensional unit vector corresponding to a point on the surface of the Bloch sphere. However since measurement of a qubit always yields only one of two possible outcomes, the mathematical representation of the quantum state of a qubit turns out to be an element of a two- (not three-) dimensional vector in a complex vector space. The state of a qubit whose polarization vector ('spin') points in a specified direction on the Bloch sphere can be written as a linear combination of a pair of 'basis states.' These basis states could for example be the standard ('computational') basis states $|0\rangle$ and $|1\rangle$

$$|\psi\rangle = \alpha|0\rangle + \beta|1\rangle, \tag{3.1}$$

where $\alpha$ and $\beta$ are complex coefficients known as *quantum amplitudes*. The mathematical space of allowed states for a qubit is a two-dimensional vector space over the field of complex numbers and is referred to as the system's Hilbert space. **The reader is urged to review the formal definition of a vector space in Appendix B. In addition, a handy summary of the equations and mathematical identities used in this chapter is presented in App. C.**

We are used to writing ordinary vectors as 'row vectors' $(x, y)$ or $(x, y, z)$ whose components are real numbers. It is traditional however in quantum mechanics to represent vectors as (possibly complex) 'column vectors.' Thus for example the standard basis states can be represented as

$$|0\rangle = \begin{pmatrix} 1 \\ 0 \end{pmatrix} \tag{3.2}$$

$$|1\rangle = \begin{pmatrix} 0 \\ 1 \end{pmatrix} \tag{3.3}$$

which gives for Eq. (3.1)

$$|\psi\rangle = \begin{pmatrix} \alpha \\ \beta \end{pmatrix}. \tag{3.4}$$

The notation $|0\rangle, |1\rangle$ refers to the direction of the polarization vector on the Bloch sphere, while the notation $0, 1$ is the computer science notation for the bit value of the standard basis states. (See Fig. 2.7.)

As described in Appendix B, it will be useful to define an 'inner product' between pairs of vectors in Hilbert space that is analogous to the dot product of two ordinary vectors[1]. The inner product of a pair of vectors is a *scalar*, that is it is an element of the field over which the vector space is defined. Ordinary vectors like the position $\vec{r}$ of a particle in three-dimensional space form a vector space over the field of real numbers and the inner (i.e., dot) product is a real number. Hilbert space is an abstract space of vectors (quantum states) over the field of complex numbers. Hence the inner product of two state vectors can be complex. Following common parlance, we will use the terms 'inner product' and 'overlap' interchangeably. The inner product between $|\psi\rangle$ and

$$|\psi'\rangle = \alpha'|0\rangle + \beta'|1\rangle = \begin{pmatrix} \alpha' \\ \beta' \end{pmatrix} \tag{3.5}$$

is defined to be (using the Dirac notation[2] discussed in Appendix B and to be explained further below)

$$\langle\psi'|\psi\rangle = \alpha'^*\alpha + \beta'^*\beta. \tag{3.6}$$

[1]Recall that the ordinary dot product of two three-dimensional vectors $\vec{v}_j = (x_j, y_j, z_j)$ is $\vec{v}_1 \cdot \vec{v}_2 = x_1x_2 + y_1y_2 + z_1z_2 = |\vec{v}_1||\vec{v}_2|\cos\theta_{12}$, where $|\vec{v}_j| = \sqrt{\vec{v}_j \cdot \vec{v}_j}$ is the length of the $j$th vector and $\theta_{12}$ is the angle between the two vectors.

[2]Mathematicians often prefer the notation $(\psi', \psi)$ for the inner product.

For ordinary vectors $\vec{A}\cdot\vec{B} = \vec{B}\cdot\vec{A}$, but notice that for a complex vector space we have to be careful about the order since

$$\langle\psi'|\psi\rangle = (\langle\psi|\psi'\rangle)^*. \tag{3.7}$$

The complex conjugation means that the inner product of any vector with itself is real

$$\langle\psi|\psi\rangle = |\alpha|^2 + |\beta|^2 = 1, \tag{3.8}$$

where the second equality follows from the Born interpretation of the magnitude squared of probability amplitudes as measurment outcome probabilities.

Dirac referred to his notation for state vectors as the 'bracket' notation. A quantum states is represented by 'ket' $|\psi\rangle$. Associated with this vector is a dual 'bra' vector defined to be the following *row* vector

$$\langle\psi| = \begin{pmatrix} \alpha \\ \beta \end{pmatrix}^\dagger = \begin{pmatrix} \alpha^* & \beta^* \end{pmatrix}, \tag{3.9}$$

where $\dagger$ indicates the adjoint, that is the complex conjugate of the transpose. Thus

$$\langle\psi|\psi'\rangle = \begin{pmatrix} \alpha^* & \beta^* \end{pmatrix} \begin{pmatrix} \alpha' \\ \beta' \end{pmatrix} = \alpha^*\alpha' + \beta^*\beta', \tag{3.10}$$

where the last equality follows from the ordinary rules of matrix multiplication (here applied to non-square matrices).

Even though the inner product can be complex, we can still think of it as telling us something about the angle between two vectors in Hilbert space. Thus for example

$$\langle 0|1\rangle = \langle\uparrow|\downarrow\rangle = 0. \tag{3.11}$$

Notice the important fact that these pairs of vectors in Hilbert space are *orthogonal* even though their corresponding qubit polarization vectors on the Bloch sphere are *co-linear* (anti-parallel) and thus *not* orthogonal in the usual geometric sense. Thus opposite points on the Bloch sphere correspond to orthogonal vectors in Hilbert space. [This is closely tied to the fact that half angles appear in Eqs. (2.16-2.15).] If two state vectors are orthogonal then the states are completely physically distinguishable. A system prepared in one of the states can be measured to uniquely determine which of the two states the system is in.

Of course if the system is in a superposition of two orthogonal states, the measurement result can be random. Conversely, if a system is in one of two states that are not orthogonal, it is not possible to reliably determine by measurement which state the system is in. We shall explore this more deeply when we discuss measurements.

## 3.1 Linear Operators on Hilbert Space

An operator defines a mapping from the Hilbert space back into the same Hilbert space. In quantum mechanics we will deal with linear operators, which for the case of a single qubit are simply 2x2 matrices that when multiplied into a column vector representing a state, produce another column vector representing another state. It turns out that physical observables in quantum mechanics are represented by Hermitian matrices. We cannot use numbers to represent physical observables because observables do not have values before we measure them. Instead observables have the potential to have different values upon measurement. It turns out that matrices can be used to represent this peculiar situation. That is matrices (since they contain many numbers) have the potential to express all the possible measurement outcomes. This takes some getting used to, so we will proceed with some examples.

We might for example wish to describe the possibility that the $z$ component of the polarization vector on the Bloch sphere could turn out, upon measurement, to be either one of the two allowed results, $+1$ or $-1$. This physical quantity (the $z$ component of the polarization vector on the Bloch sphere) is represented by the following linear operator (matrix)

$$\sigma^z = Z = \begin{pmatrix} 1 & 0 \\ 0 & -1 \end{pmatrix}. \tag{3.12}$$

where $\sigma^z$ is the standard physics notation (associated with the $z$ component of an electron spin vector say) and $Z$ is the standard quantum computer science notation. We shall use these interchangeably. [Please note that the superscript $z$ is not an exponent but simply a label. In Exercise 3.4 we define three so-called Pauli matrices labeled $x, y, z$ as the three components of a vector whose components are 2x2 matrices.]

Now notice the following interesting properties of this matrix

$$\begin{aligned}\sigma^z|0\rangle &= \begin{pmatrix} 1 & 0 \\ 0 & -1 \end{pmatrix}\begin{pmatrix} 1 \\ 0 \end{pmatrix} \\ &= \begin{pmatrix} 1 \\ 0 \end{pmatrix} = (+1)|0\rangle \qquad (3.13) \\ \sigma^z|1\rangle &= \begin{pmatrix} 1 & 0 \\ 0 & -1 \end{pmatrix}\begin{pmatrix} 0 \\ 1 \end{pmatrix} \\ &= \begin{pmatrix} 0 \\ -1 \end{pmatrix} = (-1)|1\rangle. \qquad (3.14)\end{aligned}$$

These equations tell us that $|0\rangle$ is an *eigenstate* of $\sigma^z$ with *eigenvalue* +1 and that $|1\rangle$ is an *eigenstate* of $\sigma^z$ with *eigenvalue* -1. The eigenvectors of an operator are those vectors that are mapped into themselves under the operation, up to a constant factor called the eigenvalue.

It turns out that the possible measurement results for some physical observable are given by the eigenvalues of the operator corresponding to that observable. Hence, we see that for the case of $\sigma^z$ the possible measurement results must be $\pm 1$ in agreement with experiment. Furthermore, it turns out that if a given quantum state is an eigenstate of a particular operator with eigenvalue $\lambda$ then the measured value of the observable will not be random, but in fact will always be exactly $\lambda$. Let us consider what happens when a state is not an eigenstate of an observable. Consider for example

$$\begin{aligned}|\psi\rangle &= \alpha|0\rangle + \beta|1\rangle \qquad (3.15) \\ \sigma^z|\psi\rangle &= +\alpha|0\rangle - \beta|1\rangle. \qquad (3.16)\end{aligned}$$

If $\alpha$ and $\beta$ are both non-zero, this is clearly not an eigenvector. But this is consistent with the fact that the measurement result will be random with probability $p_0 = |\alpha|^2$ of being +1 and probability $p_1 = |\beta|^2$ of being -1. The reader is urged to carefully study Box 3.1 since the topic of random measurement results and state collapse can be confusing to beginners. A common confusion among students is the idea that measurement of $\sigma^z$ is represented mathematical by acting with $\sigma^z$ on the state. This is *incorrect.*

**Box 3.1. Eigenvalues, Eigenvectors and Measurements** *Measurement of the value of a physical observable always yields one of the eigenvalues of the operator that represents that observable.* It is important to note that the act of measurement of an observable is *not* simply represented by multiplying the quantum state $|\psi\rangle$ by the operator corresponding to the observable being measured (e.g. $\sigma^z$ as in Eq. (3.16)). In particular, $\sigma^z|\psi\rangle$ is *not* the state of the system after the measurement of $\sigma^z$. Instead, the state always randomly collapses to one of the eigenstates of the operator being measured. Furthermore, the measurement result is always one of the eigenvalues of the operator being measured and the particular state to which the system collapses is the eigenvector corresponding to the eigenvalue that is the measurement result. Thus if the result of the measurement of $\sigma^z$ is $+1$, the state collapses to

$$\alpha|0\rangle + \beta|1\rangle \longrightarrow |0\rangle, \tag{3.17}$$

and the quantum amplitudes $\alpha, \beta$ are permanently 'lost.' Conversely, if the result of the measurement is $-1$, the state collapses to $|1\rangle$. From the Born rule, the probabilities that the state collapses to $|0\rangle$ and $|1\rangle$ are respectively

$$p_0 = |\alpha|^2 = \langle\psi|0\rangle\langle 0|\psi\rangle \tag{3.18}$$

$$p_1 = |\beta|^2 = \langle\psi|1\rangle\langle 1|\psi\rangle. \tag{3.19}$$

How do we know that measurement collapses the state onto one of the eigenvectors of the operator being measured? This follows from the experimental fact that a $Z$ measurement (say) that gives a certain result, will give exactly the same result if we make a second $Z$ measurement. The most general possible state is a superposition of the two eigenstates of $\sigma^z$ of the form $|\psi\rangle = \alpha|0\rangle + \beta|1\rangle$. In order for the second $Z$ measurement to not be random, we must have (from the Born rule) that $|\alpha|^2$ and $|\beta|^2$ cannot both be non-zero. We have to be in one eigenstate or the other so that the probability of getting the same measurement result is 100%. Thus the first $Z$ measurement has to collapse the state onto one of the eigenstates of $Z$.

Shortly, we will encounter other Hermitian operators that are not diagonal in the standard basis. In order to apply the Born rule for them, it is essential to re-express the state in the basis of eigenstates of the operator being measured. See Box 3.3 for discussion of this point.

As discussed further in Box 3.1, measurement collapses the state, so if we want to know the average measurement result for the state we have to prepare many copies and measure each copy *only once.* The average measurement result will of course be $(+1)p_0 + (-1)p_1 = |\alpha|^2 - |\beta|^2$. Let us compare this quantity to the so-called 'expectation value' of the operator $\sigma^z$ in the state $|\psi\rangle$ which is defined by the expression $\langle\psi|\sigma^z|\psi\rangle$. What we mean by this expression is compute the state $\sigma^z|\psi\rangle$ and then takes its inner product ('overlap') with $|\psi\rangle$:

$$\begin{aligned}\langle\psi|\sigma^z|\psi\rangle &= (\alpha^*\langle 0| + \beta^*\langle 1|)(+\alpha|0\rangle - \beta|1\rangle \\ &= +|\alpha|^2\langle 0|0\rangle - \alpha^*\beta\langle 0|1\rangle + \beta^*\alpha\langle 1|0\rangle - |\beta|^2\langle 1|1\rangle \\ &= |\alpha|^2 - |\beta|^2. \qquad (3.20)\end{aligned}$$

Or equivalently in matrix notation the 'expectation value of the observable $\sigma^z$ in the state $|\psi\rangle$' is given by

$$\begin{aligned}\langle\psi|\sigma^z|\psi\rangle &= \begin{pmatrix} \psi_0^* & \psi_1^* \end{pmatrix} \begin{pmatrix} 1 & 0 \\ 0 & -1 \end{pmatrix} \begin{pmatrix} \psi_0 \\ \psi_1 \end{pmatrix} \qquad (3.21) \\ &= |\psi_0|^2 - |\psi_1|^2 = |\alpha|^2 - |\beta|^2. \qquad (3.22)\end{aligned}$$

*Thus we have the nice result that the average measurement result for some observable is simply the expectation value of the observable (operator) in the quantum state being measured.* We see again that the operator associated with a physical observable contains information about all possible values that could ever be measured for that observable. This is why the operator has to be a matrix. We again emphasize however that individual measurement results are random and the state after the measurement collapses to the eigenvector corresponding to the measured eigenvalue. The state after the measurement has nothing to do with $\sigma^z|\psi\rangle$.

Can we now say something about how random the measurement result will be? Let us begin by reminding ourselves about some basic facts about classical random variables. The reader is urged to review the discussion of probability and statistics in Appendix A. Suppose some random variable $\xi$ takes on values from the set $\{v_1, v_2, \ldots, v_M\}$ and value $v_j$ occurs with probability $p_j$. Then the *mean value* (also known as the *expectation value*) is given by

$$\bar{\xi} = \sum_{j=1}^{M} p_j v_j. \qquad (3.23)$$

where the overbar indicates statistical average. One measure of the randomness of $\xi$ is its variance defined by

$$\mathrm{Var}(\xi) \equiv \overline{(\xi - \bar{\xi})^2} = \overline{\xi^2} - \bar{\xi}^2 \tag{3.24}$$

$$= \sum_{j=1}^{M} p_j v_j^2 - \left[\sum_{j=1}^{M} p_j v_j\right]^2 . \tag{3.25}$$

We see that the variance is the mean square deviation of the measured quantity from the average and so is a measure of the size of the random fluctuations in the $\xi$.

**Exercise 3.1.** Statistical Variance

a) Derive Eq. (3.25).

b) Assuming that all the eigenvalues $v_j$ are distinct (i.e. non-degenerate), prove that the variance vanishes if and only if one of the $p_j$'s is unity and the rest vanish (so that $\xi$ is not random). If one of the eigenvalues is $m$-fold degenerate, then the corresponding $m$ probabilities must add up to unity and the rest must vanish.

Let us now turn to the quantum case. Let observable $Q$ have eigenvalues $\{v_1, v_2, \ldots, v_M\}$. (So far we have only considered the case of a single qubit for which $M = 2$. In more general cases (e.g. multiple qubits), the Hilbert space dimension $M$ can be larger.) The analog of Eq. (3.25) for the variance of the measurement results for $Q$ in state $|\psi\rangle$ is

$$\mathrm{Var}(Q) = \langle\psi|Q^2|\psi\rangle - \langle\psi|Q|\psi\rangle^2. \tag{3.26}$$

Now notice that if $|\psi\rangle$ is an eigenvector of $Q$ (say the $m$th eigenvector)

$$Q|\psi\rangle = v_m|\psi\rangle, \tag{3.27}$$

then

$$\mathrm{Var}(Q) = v_m^2\langle\psi|\psi\rangle - [v_m\langle\psi|\psi\rangle]^2 = 0. \tag{3.28}$$

Thus only superpositions of states (in the basis of eigenvectors appropriate to the measurement!) with different eigenvalues will have randomness in the measurement results. This is entirely consistent with the experimental facts described above for a single qubit.

**Exercise 3.2.** Since the eigenvalues of $\sigma^z$ are $\pm 1$, we expect that $(\sigma^z)^2$ has both of its eigenvalues equal to unity. From the matrix form of $\sigma^z$ prove that its square is the identity matrix

$$(\sigma^z)^2 = \hat{I} = \begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix}.$$

**Box 3.2. Observables as Hermitian operators** Hermitian matrices (that is, matrices obeying $M^\dagger = M$, where again $\dagger$ indicates transpose followed by complex conjugation) are guaranteed to have real eigenvalues and guaranteed that their eigenvectors form a complete basis set that spans the Hilbert space (more on what this means later; see also Appendix B). Since the measured values of physical observables are always real, physical observables are always represented by Hermitian matrices.

Let us now demonstrate that Hermitian matrices have real eigenvalues and that their eigenvectors are orthogonal. Consider the $j$th and $k$th (normalized) eigenvectors

$$\begin{aligned} H|\psi_j\rangle &= \lambda_j|\psi_j\rangle, && (3.29)\\ H|\psi_k\rangle &= \lambda_k|\psi_j\rangle. && (3.30)\end{aligned}$$

Taking the adjoint of the second equation we obtain

$$\langle\psi_k|H^\dagger = \lambda_k^*\langle\psi_k|. \qquad (3.31)$$

Assuming $H^\dagger = H$, can now use Eq. (3.29) and Eq. (3.31) to obtain

$$\begin{aligned} \langle\psi_k|H|\psi_j\rangle &= \lambda_j\langle\psi_k|\psi_j\rangle, && (3.32)\\ \langle\psi_k|H|\psi_j\rangle &= \lambda_k^*\langle\psi_k|\psi_j\rangle. && (3.33)\end{aligned}$$

Subtracting these yields

$$0 = (\lambda_j - \lambda_k^*)\langle\psi_k|\psi_j\rangle. \qquad (3.34)$$

For the case $j = k$, we know that (by construction) $\langle\psi_j|\psi_j\rangle = 1$, and hence the imaginary part of $\lambda_j$ must vanish. Thus all the eigenvalues of an Hermitian operator are real. If for $j \neq k$, the eigenvalues are non-degenerate (i.e., $\lambda_j \neq \lambda_k$), then Eq. (3.34) requires $\langle\psi_k|\psi_j\rangle = 0$ and so the two eigenvectors must be orthogonal. Thus if the full spectrum is non-degenerate, the set of eigenvectors is orthonormal: $\langle\psi_k|\psi_j\rangle = \delta_{kj}$ (where $\delta_{kj}$ is the Kronecker delta symbol which vanishes if $k \neq j$ and is unity for $k = j$).

If $M$ eigenvalues are degenerate, then any superposition of the $M$ eigenvectors is also an eigenvector. If they are not orthogonal, they can be made orthogonal by taking appropriate linear combinations of the set of $M$ eigenvectors (via the so-called Gram-Schmidt procedure).

## 3.2 Dirac Notation for Operators

We have seen that in Dirac's bracket notation, the inner product $\langle\Phi|\Psi\rangle$ is a (possibly complex) number. As discussed in Appendix B, the so-called *outer product* $G = |\Psi\rangle\langle\Phi|$ turns out to be an operator on the space. To see why, simply operate with $G$ on an arbitrary state $|\chi\rangle$

$$G|\chi\rangle = (|\Psi\rangle\langle\Phi|)\,|\chi\rangle = |\Psi\rangle\,(\langle\Phi|\chi\rangle) = g|\Psi\rangle, \tag{3.35}$$

where $g$ is the complex number representing the inner product of $|\Phi\rangle$ and $|\Psi\rangle$,

$$g = \langle\Phi|\chi\rangle. \tag{3.36}$$

Hence when applied to any vector in the Hilbert space, $G$ returns another vector in the Hilbert space. Thus it is an operator and indeed it is a linear operator since

$$G(\alpha|\chi\rangle + \beta|\phi\rangle) = \alpha G|\chi\rangle + \beta G|\phi\rangle. \tag{3.37}$$

As a specific example, consider the operator

$$G = |0\rangle\langle 0| - |1\rangle\langle 1|. \tag{3.38}$$

Clearly

$$\begin{aligned} G|0\rangle &= (+1)|0\rangle && (3.39)\\ G|1\rangle &= (-1)|1\rangle, && (3.40) \end{aligned}$$

so it must be that $G = \sigma^z$. To confirm this let us write $G$ out in matrix form

$$\begin{aligned} G &= \begin{pmatrix} 1 \\ 0 \end{pmatrix} \begin{pmatrix} 1 & 0 \end{pmatrix} - \begin{pmatrix} 0 \\ 1 \end{pmatrix} \begin{pmatrix} 0 & 1 \end{pmatrix} && (3.41)\\ &= \begin{pmatrix} 1 & 0 \\ 0 & 0 \end{pmatrix} - \begin{pmatrix} 0 & 0 \\ 0 & 1 \end{pmatrix} && (3.42)\\ &= \begin{pmatrix} 1 & 0 \\ 0 & -1 \end{pmatrix}, && (3.43) \end{aligned}$$

where the last result follows from the rules of matrix products given in App. B and agrees with Eq. (3.12).

The above result is a particular case of the general fact that in Dirac's notation, Hermitian operators take on a very simple form

$$V = \sum_{j=1}^{M} v_j |\psi_j\rangle\langle\psi_j|, \tag{3.44}$$

where $|\psi_j\rangle$ is the $j$th eigenvector of $V$ with eigenvalue $v_j$.

**Exercise 3.3.** Show that the representation of the operator $V$ in Eq. (3.44) correctly reproduces the required property of $V$ that

$$V|\psi_m\rangle = v_m|\psi_m\rangle,$$

for every eigenvector $|\psi_m\rangle$ of $V$.

**Exercise 3.4.** The three components of the (dimensionless) spin (qubit) polarization vector $\vec{\sigma} = (\sigma^x, \sigma^y, \sigma^z)$ are each physical observables. The $2 \times 2$ matrices representing these operators are known as the Pauli matrices. We have already seen the matrix representation for $\sigma^z$ in the up/down ($Z$) basis

$$\sigma^z = \begin{pmatrix} +1 & 0 \\ 0 & -1 \end{pmatrix}. \tag{3.45}$$

Using the known form of the states $|\pm X\rangle$ from Eq. (2.18) and Eq. (2.19) and $|\pm Y\rangle$ in Eq. (2.20), show that in the $Z$ basis

$$\sigma^x = \begin{pmatrix} 0 & +1 \\ +1 & 0 \end{pmatrix} \tag{3.46}$$

$$\sigma^y = \begin{pmatrix} 0 & -i \\ +i & 0 \end{pmatrix}. \tag{3.47}$$

## 3.3 Orthonormal bases for qubit states

Using the standard state parametrization in Eqs. (2.16-2.15), it is possible to show that the quantum states $|\pm\hat{n}\rangle$ corresponding to any pair of oppositely directed unit vectors $\pm\hat{n}$ on the Bloch sphere are orthogonal

$$\langle -\hat{n}| + \hat{n}\rangle = 0. \tag{3.48}$$

Since were are dealing with a Hilbert space of only two dimensions and we have an orthonormal pair of states in the Hilbert space, they constitute a complete basis for expressing any vector in the Hilbert space.

Similarly, it is straightforward to derive the so-called *completeness relation*

$$|+\hat{n}\rangle\langle+\hat{n}| + |-\hat{n}\rangle\langle-\hat{n}| = \hat{I}, \tag{3.49}$$

where $\hat{I}$ is the identity operator whose matrix representation is

$$\hat{I} = \begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix}. \tag{3.50}$$

This result is the simplest illustration of the theorem that the eigenvectors of any Hermitian operator on a Hilbert space form a complete basis for that space. Furthermore, if the spectrum is non-degenerate (no two eigenvalues are the same), then the eigenvectors are automatically orthogonal and thus form a complete orthonormal basis. See Box 3.2 and App. B.

**Exercise 3.5.** Derive Eq. (3.48).

**Exercise 3.6.** Derive Eq. (3.49).

The completeness relation is a powerful result because it allows us to express any state $|\psi\rangle$ in any basis by simply writing

$$|\psi\rangle = \hat{I}|\psi\rangle = |+\hat{n}\rangle\langle+\hat{n}|\psi\rangle + |-\hat{n}\rangle\langle-\hat{n}|\psi\rangle. \tag{3.51}$$

We see immediately the coefficients in the expansion of the state $|\psi\rangle$ in the basis are simply computed as the inner products of the basis vectors with the state $|\psi\rangle$.

The results above are very reminiscent of how we find the representation of an ordinary vector, say a position vector in 2D. We may have some orthogonal basis vectors for our coordinate system, for example $\hat{x}, \hat{y}$, or a rotated set $\hat{\imath}, \hat{\jmath}$. We can represent any vector as

$$\begin{aligned} \vec{r} &= (r_x, r_y) = \hat{x}(\hat{x}\cdot\vec{r}) + \hat{y}(\hat{y}\cdot\vec{r}) && (3.52)\\ &= \hat{\imath}(\hat{\imath}\cdot\vec{r}) + \hat{\jmath}(\hat{\jmath}\cdot\vec{r}). && (3.53)\end{aligned}$$

This suggests we can think of the identity transformation as

$$\hat{I} = \hat{x}\hat{x} + \hat{y}\hat{y} = \hat{\imath}\hat{\imath} + \hat{\jmath}\hat{\jmath}, \tag{3.54}$$

where we interpret $\hat{x}\hat{x}$ applied to a vector $\hat{V}$ to mean $\hat{x}(\hat{x}\cdot\vec{V})$. Note the similarity between Eq. (3.54) and Eq. (3.49).

It is useful at this point to discuss the concepts of *projections* and *projectors*. Recall that when we project an ordinary vector $\vec{r}$ onto the $x$ axis, we simply remove all components of $\vec{r}$ other than the $x$ component. Thus the projection of $\vec{r}$ onto the $x$ axis is the vector $\hat{x}V_x = \hat{x}(\hat{x}\cdot\vec{V})$. We can think of $P_x = \hat{x}\hat{x}$ as a *projection operator* (or projector) because

$$P_x\vec{V} = (\hat{x}\hat{x})\vec{V} = \hat{x}(\hat{x}\cdot V). \tag{3.55}$$

Notice that $P_x$ satisfies the defining characteristic of projection operators

$$(P_x)^2 = P_x. \tag{3.56}$$

This has a simple interpretation–once the vector is projected onto the $x$ axis, further projection doesn't do anything.

We can think of the shadow of an object cast onto the ground as the projection of the objection onto the horizontal ($xy$) plane. This is accomplished by the projection operator

$$P_{xy} = \hat{x}\hat{x} + \hat{y}\hat{y} = \hat{I} - \hat{z}\hat{z}, \tag{3.57}$$

where $\hat{I} = \hat{x}\hat{x} + \hat{y}\hat{y} + \hat{x}\hat{z}$ is the identity for the 3D case. This shows us that projection onto the $xy$ plane simply removes the component of the vector normal to the plane. Despite this more complicated form, it is straightforward to show that $(P_{xy})^2 = P_{xy}$ as required. The key to this is that the basis vectors $\hat{x}, \hat{y}, \hat{z}$ are orthonormal.

The analogous projector onto the vector $|\hat{n}\rangle$ in the Hilbert space describing the polarization of a qubit is simply

$$P_{\hat{n}} = |\hat{n}\rangle\langle\hat{n}|. \tag{3.58}$$

It is straightforward to show that this is idempotent $P_{\hat{n}}P_{\hat{n}} = P_{\hat{n}}$ as required.

**Box 3.3. Measurements not in the standard basis** Suppose we are given the (standard basis) state

$$|\Psi\rangle = \alpha|0\rangle + \beta|1\rangle, \tag{3.59}$$

and asked to measure an Hermitian operator

$$\mathcal{M} = M_+|\hat{n}\rangle\langle\hat{n}| + M_-|-\hat{n}\rangle\langle-\hat{n}| \tag{3.60}$$

that is not diagonal in the standard basis (i.e., $\hat{n} \neq \pm\hat{z}$). Clearly the two eigenvalues and eigenvectors of $\mathcal{M}$ are given by

$$\mathcal{M}|+\hat{n}\rangle = M_+|+\hat{n}\rangle \tag{3.61}$$
$$\mathcal{M}|-\hat{n}\rangle = M_-|-\hat{n}\rangle. \tag{3.62}$$

In order to apply the Born rule, it is essential that we express the state in the basis of the eigenstates of $\mathcal{M}$

$$|\Psi\rangle = \alpha'|+\hat{n}\rangle + \beta'|-\hat{n}\rangle, \tag{3.63}$$

from which we determine that measurement result $M_+$ will occur with probability $P_+ = |\alpha'|^2$ and the state will collapse in this case to $|+\hat{n}\rangle$, while measurement result $M_-$ will occur with probability $P_- = |\beta'|^2$ and the state will collapse in this case to $|-\hat{n}\rangle$.
We can use the completeness relation in Eq. (3.49) to change the basis

$$|\Psi\rangle = |+\hat{n}\rangle\langle+\hat{n}|\Psi\rangle + |-\hat{n}\rangle\langle-\hat{n}|\Psi\rangle \tag{3.64}$$
$$\alpha' = \langle+\hat{n}|\Psi\rangle = \alpha\langle+\hat{n}|0\rangle + \beta\langle+\hat{n}|1\rangle \tag{3.65}$$
$$\beta' = \langle-\hat{n}|\Psi\rangle = \alpha\langle-\hat{n}|0\rangle + \beta\langle-\hat{n}|1\rangle. \tag{3.66}$$

Thus, Eq. (3.49) along with the Born rule tells us that means that, given an arbitrary state $|\psi\rangle$, a measurement asking the question 'Is the state $|\Psi\rangle$ actually $|\hat{n}\rangle$' will be answered 'yes' with probability given by $|\alpha'|^2 = \langle\psi|P_{\hat{n}}|\psi\rangle = |\langle+\hat{n}|\psi\rangle|^2$. Correspondingly the question 'Is the state $|\Psi\rangle$ actually $|+\hat{n}\rangle$' will be answered 'yes' with probability given by $|\beta'|^2 = \langle\psi|P_{-\hat{n}}|\psi\rangle = |\langle-\hat{n}|\psi\rangle|^2$.

**Exercise 3.7.** Show that $| \pm \hat{n}\rangle$ is an eigenvector of

$$\Gamma_{\hat{n}} = \hat{n} \cdot \vec{\sigma} = (\hat{n} \cdot \hat{x})\sigma^x + (\hat{n} \cdot \hat{y})\sigma^y + (\hat{n} \cdot \hat{z})\sigma^z \tag{3.67}$$

with eigenvalue $\pm 1$

$$\Gamma_{\hat{n}}| + \hat{n}\rangle = (+1)| + \hat{n}\rangle \tag{3.68}$$
$$\Gamma_{\hat{n}}| - \hat{n}\rangle = (-1)| - \hat{n}\rangle. \tag{3.69}$$

Hint: Show that

$$\Gamma_{\hat{n}} = | + \hat{n}\rangle\langle\hat{n}| - | - \hat{n}\rangle\langle -\hat{n}|. \tag{3.70}$$

This is simply a reflection of the rotation invariance of space. We can align our measurement apparatus to measure the qubit polarization in the arbitrary direction $\hat{n}$ and the state of the qubit will collapse to either $| + \hat{n}\rangle$ or $| - \hat{n}\rangle$.

## 3.3.1 Gauge Invariance

We have seen that any two states at opposite points on the Bloch sphere form an orthonormal basis that can be used to represent any state in the 2D Hilbert space of a single qubit. One confusing aspect of all this that we have not yet discussed, is the following. The states $|\pm\hat{n}\rangle$ in the Hilbert space corresponding to the Bloch sphere unit vectors $\pm\hat{n}$ are *not* unique. Each basis vector can be multiplied by a different arbitrary phase factor $|v_+\rangle = e^{i\xi_+}|+\hat{n}\rangle$ and $|v_-\rangle = e^{i\xi_-}| - \hat{n}\rangle$ and we would still have a perfectly good orthonormal basis obeying

$$\langle v_+|v_+\rangle = \langle v_-|v_-\rangle = 1, \tag{3.71}$$
$$\langle v_-|v_+\rangle = 0, \tag{3.72}$$

and the completeness relation

$$\hat{I} = |v_+\rangle\langle v_+| + |v_-\rangle\langle v_-|. \tag{3.73}$$

The particular choice of phase factors cancels out in the above expressions. The specific representation of one other operator (besides the identity) is also independent of the so-called 'gauge' choice that we make by picking particluar phase factors. For example, the (diagonal) operator that is measured by an apparatus that detects the $\hat{n}$ component of the qubit polarization vector

$$\begin{aligned}\hat{n}\cdot\sigma &= |v_+\rangle\langle v_+| - |v_-\rangle\langle v_-| &(3.74)\\ &= |+\hat{n}\rangle\langle +\hat{n}| - |-\hat{n}\rangle\langle -\hat{n}|, &(3.75)\end{aligned}$$

is invariant and the probability of the two measurement results is unaffected by the gauge choice. However non-diagonal operators such as the spin flip operator

$$\begin{aligned}\hat{Q} &= |v_+\rangle\langle v_-| + |v_-\rangle\langle v_+| &(3.76)\\ &= e^{+i(\xi_+-\xi_-)}|+\hat{n}\rangle\langle -\hat{n}| + e^{-i(\xi_+-\xi_-)}|-\hat{n}\rangle\langle +\hat{n}| &(3.77)\end{aligned}$$

are gauge dependent when expressed in the $|\pm\hat{n}\rangle$ basis. Of course the state vectors also become gauge dependent

$$\begin{aligned}|\psi\rangle &= \alpha|v_+\rangle + \beta|v_-\rangle &(3.78)\\ &= \alpha e^{i\xi_+}|+\hat{n}\rangle + \beta e^{i\xi_-}|-\hat{n}\rangle, &(3.79)\end{aligned}$$

and so nothing changes in the physics.

Interestingly, even the apparently fixed definition for $|+\hat{n}\rangle$ in Eqs. (2.16-2.16) is ambiguous when it comes to writing down the other basis vector $|-\hat{n}\rangle$. If $\hat{n} = (\sin\theta\cos\varphi, \sin\theta\sin\varphi, \cos\theta)$, then we can arrive at the oppositely directed unit vector $-\hat{n}$ via two different routes: i) $\theta\to\theta+\pi, \varphi\to\varphi$, or ii) $\theta\to\pi-\theta, \varphi\to\varphi+\pi$. Consider for example the case $\theta=\pi/2, \varphi=0$ which gives

$$|\hat{n}\rangle = |+X\rangle = \frac{1}{\sqrt{2}}\left[|0\rangle + |1\rangle\right]. \qquad (3.80)$$

Both transformation (i) and (ii) take $\hat{n}$ to $-\hat{n}$ on the Bloch sphere but the two resulting states in the Hilbert space differ by a gauge choice (i.e., a phase)

$$\begin{aligned}|-\hat{n}\rangle = |-X\rangle &= -\frac{1}{\sqrt{2}}\left[|0\rangle - |1\rangle\right] \qquad \text{Method(i)} &(3.81)\\ &= +\frac{1}{\sqrt{2}}\left[|0\rangle - |1\rangle\right] \qquad \text{Method(ii)} &(3.82)\end{aligned}$$

Method (ii) self-consistently keeps $\theta$ in the range $0\le\theta\le\pi$ and is the standard gauge choice. Going back to Eq. (2.19) and Eq. (2.20), we see that Method (ii) was used in to define $|-X\rangle$ and $|-Y\rangle$ and yields the standard form of the Pauli matrices displayed in Ex. 3.4.

## 3.4 Rotations in Hilbert Space

In order to build a quantum computer we need to have complete control over the quantum states of a system of many qubits. For the moment however let us consider how we might create an arbitrary state of a single qubit starting from some fiducial state, typically taken to be either $|0\rangle$ or $|1\rangle$ in the standard basis (also known as the computational basis). Since the most general state is a superposition state $|\hat{n}\rangle$ corresponding to an arbitrary point $\hat{n}$ on the Bloch sphere, we need to be able to perform rotations in Hilbert space that correspond to the rotation of the Bloch vector some initial position to any desired position on the Bloch sphere. Being able to prepare arbitrary states is a requirement for carrying out a quantum computation.

Because every state in the single- or multi-qubit Hilbert space has the same length $\langle\psi|\psi\rangle = 1$, rotations are the only operations we need to move throughout the entire Hilbert space. There is (despite our earlier discussion of interference from adding quantum amplitudes in Eq. (2.13)) no concept of 'translations' in Hilbert space. If we add a vector $|\phi\rangle$ to the vector $|\psi\rangle$ we of course obtain another element of the complex vector space, but generically $|\psi\rangle + |\phi\rangle$ is not properly normalized to unity. One can show that the normalized version of the state

$$|\Lambda\rangle = \frac{1}{\sqrt{W}}\left(|\psi\rangle + |\phi\rangle\right), \tag{3.83}$$

with

$$W = 2 + \langle\psi|\phi\rangle + \langle\phi|\psi\rangle, \tag{3.84}$$

can be obtained from the initial state $|\psi\rangle$ via a rotation. The idea is illustrated schematically for ordinary 2D vectors in Fig. 3.1.

It turns out that rotations in Hilbert space are executed via unitary operations. A unitary matrix $U$ obeys two defining conditions

$$U^\dagger U = I, \ (U \text{ is an isometry}) \tag{3.85}$$

$$UU^\dagger = I. \ (U \text{ is a coisometry}) \tag{3.86}$$

where $I$ is the identity matrix. Thus, both the left and the right inverse of $U$ is simply the adjoint $U^{-1} = U^\dagger$. It turns out that unitary transformations preserve the inner products between vectors in Hilbert space, just as the

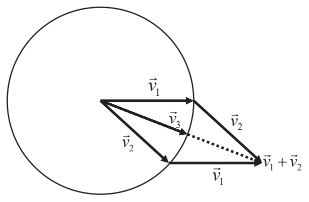

Figure 3.1: Two unit vectors $\vec{v}_1, \vec{v}_2$ whose end points lie on the unit circle in 2D. The sum of the vectors $\vec{v}_1 + \vec{v}_2$ (generically) has length squared $W = 2 + 2(\vec{v}_1 \cdot \vec{v}_2) \neq 1$. The normalized vector $\vec{v}_3 = \frac{1}{\sqrt{W}}(\vec{v}_1 + \vec{v}_2)$ does lie on the unit circle and can be obtained by a rotation applied to either $\vec{v}_1$ or $\vec{v}_2$.

more familiar orthogonal rotation matrices preserve the angles (dot products) between ordinary vectors.[3]

To better understand rotations consider the representation of an arbitrary state vector in terms of some orthonormal basis $\{|j\rangle; j = 1, 2, 3, \ldots, N\}$ spanning the $N$-dimensional Hilbert space

$$|\psi\rangle = \sum_{j=0}^{N} \psi_j |j\rangle. \tag{3.87}$$

From the Born rule, the probability of measuring the system to be in the basis state $|j\rangle$ is given by $p_j = |\psi_j|^2$. The requirement that the total probability be unity gives us the normalization requirement on the state vector

$$\sum_{j=1}^{N} |\psi_j|^2 = 1. \tag{3.88}$$

Now consider a linear operation $U$ that preserves the length of every vector

[3]Recall that the defining property of an orthogonal matrix is that $R^{-1} = R^{\mathrm{T}}$. Thus a unitary matrix whose elements are all real is an orthogonal matrix. Unitary matrices are the natural generalization of orthogonal rotation matrices to vector spaces over the field of complex numbers. This point is discussed further below in the vicinity of Eq. (3.129).

in the Hilbert space.

$$|\psi'\rangle = U|\psi\rangle = \sum_{j=1}^{N} \psi_j U|j\rangle, \tag{3.89}$$

$$\langle\psi'|\psi'\rangle = \langle\psi|U^\dagger U|\psi\rangle \tag{3.90}$$

$$= \sum_{j,k=1}^{N} \psi_k^* \psi_j \langle k|U^\dagger U|j\rangle \tag{3.91}$$

$$= \sum_{j=1}^{N} |\psi_j|^2 \langle j|U^\dagger U|j\rangle + \sum_{j\neq k}^{N} \psi_k^* \psi_j \langle k|U^\dagger U|j\rangle. \tag{3.92}$$

Since $U$ preserves the length of every vector we have

$$\langle\psi|U^\dagger U|\psi\rangle = 1 \tag{3.93}$$

and

$$\langle j|U^\dagger U|j\rangle = 1 \tag{3.94}$$

for every basis vector $|j\rangle$. Thus we must have

$$\sum_{j\neq k}^{N} \psi_k^* \psi_j \langle k|U^\dagger U|j\rangle = 0 \tag{3.95}$$

for all possible choices of the set of amplitudes $\psi_j; j = 1, 2, 3, \ldots, N$. Thus it must be that

$$\langle k|U^\dagger U|j\rangle = 0 \tag{3.96}$$

for $k \neq j$ and hence

$$\langle k|U^\dagger U|j\rangle = \delta_{kj}, \tag{3.97}$$

where the Kronecker delta symbol $\delta_{jk} = 1$ for $j = k$ and 0 for $j \neq k$. Equivalently, this means

$$U^\dagger U = I. \tag{3.98}$$

Thus the only linear operations that conserve probability for all states are unitary operations. It follows that unitary transformations preserve not only the inner product of states with themselves but also preserve the inner products between any pair of states

$$|\phi'\rangle = U|\phi\rangle \tag{3.99}$$
$$|\psi'\rangle = U|\psi\rangle \tag{3.100}$$
$$\langle\phi'|\psi'\rangle = \langle\phi|U^\dagger U\psi\rangle = \langle\phi|\psi\rangle. \tag{3.101}$$

It turns out that in an ideal, dissipationless closed quantum system, the evolution of the system from its initial state at time 0 to its final state at time $t$ is described by a unitary transformation. We can control the time evolution, and thus create different unitary operations, by applying control signals to our quantum system. The specifics of the physics of how this is done for different systems using laser beams, microwave pulses, magnetic fields, etc. will not concern us here. We will for now simply postulate that the only operations available to us to control the quantum system are multiplication of the starting state by a unitary matrix to effect a rotation in Hilbert space. This seems reasonable because we are required to conserve total probability.

It turns out that in a deep sense, unitary transformations preserve information. In a so-called open quantum system that is coupled to its environment (also called a 'bath'), the time evolution of the system plus bath is unitary but the evolution of the system alone is not because information about the system can leak into the environment and is no longer conserved (assuming we cannot access it once it is in the bath).

Being able to rotate states in Hilbert space can also be very useful for the purposes of measurement. It often happens that the energy eigenstates of the system (i.e., of the Hamiltonian operator) constitute the only basis in which measurements can be conveniently made. Typically we represent the energy eigenstates in terms of the standard basis states $|0\rangle$ and $|1\rangle$. Thus the Hamiltonian (energy operator) is

$$H = \frac{E_0 + E_1}{2}\hat{I} + \frac{E_1 - E_0}{2}\sigma^z, \tag{3.102}$$

which has eigenvalues $E_0$ and $E_1$. If we choose the zero of energy to be half way between the ground and excited state energies then $E_0 + E_1 = 0$ and we can drop the first term. If we are able to measure the energy (say) then the preferred measurement operator to which we have access is $\sigma^z$. If the qubit is in the state

$$|\psi\rangle = \alpha|0\rangle + \beta|0\rangle, \tag{3.103}$$

and we are able to prepare many copies of this state, a histogram of the measurement results $Z = \pm 1$ plus the Born rule allows us to estimate the values of $|\alpha|^2$ and $|\beta|^2$. We cannot however deduce the relative complex phase of $\alpha$ and $\beta$. (Recall that WLOG we can take $\alpha$ to be real.) To fully determine the state we need (in general) to be able to measure all three components of the 'spin' vector $\vec{\sigma}$. If we only have access to measurements of $\sigma^z$, then full state 'tomography' seems to be impossible. However if we prepend certain selected rotations of the state before making making the $Z$ measurement we can achieve our goal. For example a rotation by $\pi/2$ around the $y$ axis takes $|+X\rangle$ to $|-Z\rangle$ and $|-X\rangle$ to $|+Z\rangle$. Similarly a rotation by $\pi/2$ around the $x$ axis takes $|+Y\rangle$ to $|+Z\rangle$ and $|-Y\rangle$ to $|-Z\rangle$. Thus we can measure all three components of the qubit polarization (spin) vector. In fact, we can rotate any state $|\hat{n}\rangle$ into $|+Z\rangle$ and thus measure the operator $\hat{n}\cdot\vec{\sigma}$.

Before we learn how to rotate states in Hilbert space, let us review the familiar concept of rotations in ordinary space. For example if we start with an ordinary 3D unit vector on the Bloch sphere

$$\hat{n} = x\hat{x} + y\hat{y} + z\hat{z} = [\sin\theta\cos\varphi\,\hat{x} + \sin\theta\sin\varphi\,\hat{y} + \cos\theta\,\hat{z}], \tag{3.104}$$

we can rotate it through an angle $\chi$ around the $z$ axis to yield a new vector

$$\hat{n}\,' = [\sin\theta\cos(\varphi+\chi)\,\hat{x} + \sin\theta\sin(\varphi+\chi)\,\hat{y} + \cos\theta\,\hat{z}], \tag{3.105}$$

which simply corresponds to a transformation of the polar coordinates $\theta \to \theta, \varphi \to \varphi + \chi$. If we choose to represent $\hat{n}$ as a column vector

$$\hat{n} = \begin{pmatrix} x \\ y \\ z \end{pmatrix} = \begin{pmatrix} \sin\theta\cos\varphi \\ \sin\theta\sin\varphi \\ \cos\theta \end{pmatrix}, \tag{3.106}$$

then, using the trigonometric identities $\cos(\theta+\chi) = \cos(\theta)\cos(\chi) - \sin(\theta)\sin(\chi)$ and $\sin(\theta+\chi) = \sin(\theta)\cos(\chi) + \cos(\theta)\sin(\chi)$, it is straightforward to show that the new vector $\hat{n}\,'$ is represented by

$$\hat{n}\,' = \begin{pmatrix} x' \\ y' \\ z' \end{pmatrix} = R_z(\chi)\begin{pmatrix} x \\ y \\ z \end{pmatrix}, \tag{3.107}$$

where $R_z(\chi)$ is a $3\times 3$ 'rotation matrix'

$$R_z(\chi) = \begin{pmatrix} \cos\chi & -\sin\chi & 0 \\ \sin\chi & \cos\chi & 0 \\ 0 & 0 & 1 \end{pmatrix}. \tag{3.108}$$

Of course, if we follow this by a rotation through angle $-\chi$ we must return the vector to its original orientation. Hence $R_z(-\chi)R_z(\chi) = \hat{I}$ or equivalently $R_z(-\chi) = R_z^{-1}(\chi)$. From the fact that sin and cos are respectively odd and even in their arguments, it is straightforward to show that

$$R_z^{-1}(\chi) = R_z^{\mathrm{T}}(\chi). \tag{3.109}$$

Matrices whose transpose is equal to their inverse are called *orthogonal*, and it turns out that rotations (for ordinary vectors) are always represented by orthogonal matrices.

Two natural properties of rotations are that they preserve the length of vectors and they preserve the angle between vectors. These facts can be summarized in the single statement that if $\vec{r}_1{}' = R_z(\chi)\vec{r}_1$ and $\vec{r}_2{}' = R_z(\chi)\vec{r}_2$, then $\vec{r}_1{}' \cdot \vec{r}_2{}' = \vec{r}_1 \cdot \vec{r}_2$. That is, the dot product between any two vectors (including the case of the the dot product of a vector with itself) is invariant under rotations. The mathematical source of the preservation of lengths can be traced back to the fact that the determininant of an orthogonal matrix is unity.

Let us now turn to rotations in Hilbert space. There, must be some connection to ordinary rotations, because rotation of the qubit spin vector on the Bloch sphere is an ordinary (3D vector) rotation. However the Hilbert space is only two-dimensional, and unlike the example above where the standard basis vectors were $\hat{x}, \hat{y}$ and $\hat{y}$, the standard basis vectors in the Hilbert space correspond to the orthogonal states $|+Z\rangle = |0\rangle$ and $|-Z\rangle = |1\rangle$. Furthermore, the inner product in Hilbert space involves complex conjugation unlike the case of the dot product for ordinary vectors. Hence we expect that rotation operations in Hilbert space will not look like 3D rotations of the unit vectors on the Bloch sphere.

Let us begin with rotations around the $z$ axis. We know from the example above, that rotation of the 3D vector $\hat{n}$ on the Bloch sphere by an angle $\chi$ around the $z$ axis, simply corresponds to the transformation $\varphi \to \varphi + \chi$. From the standard quantum state representation in Eqs. (2.16-2.15), we see that this simply changes the relative phase of the coefficients of $|0\rangle$ and $|1\rangle$. Let us therefore consider the following operator on the Hilbert space which does something very similar

$$U_z(\chi) = e^{-i\frac{\chi}{2}\sigma^z}. \tag{3.110}$$

What do we mean by the exponential of a matrix? One way to interpret this expression is via a power series expansion

$$U_z(\chi) = \sum_{n=0}^{\infty} \frac{1}{n!} \left[-i\frac{\chi}{2}\right]^n [\sigma^z]^n. \tag{3.111}$$

The series should converge provided that $\frac{|\chi|}{2}|\sigma^z|$ lies within the radius of convergence of the series expansion. Here $|\sigma^z|$ means the absolute value of the largest eigenvalue of the matrix. Since $|\sigma^z| = 1$ and since the exponential is an entire function (it is analytic everywhere in the complex plane and hence the series has infinite radius of convergence), the series does converge[4].

Using the result of Ex. 3.2, we can rewrite Eq. (3.111) as

$$\begin{aligned} U_z(\chi) &= e^{-i\frac{\chi}{2}\sigma^z} = \sum_{n=\text{ even}} \frac{1}{n!} \left[-i\frac{\chi}{2}\right]^n \hat{I} + \sum_{n=\text{ odd}} \frac{1}{n!} \left[-i\frac{\chi}{2}\right]^n \sigma^z \\ &= \cos\left(\frac{\chi}{2}\right) \hat{I} - i\sin\left(\frac{\chi}{2}\right) \sigma^z. \end{aligned} \tag{3.112}$$

[4]For infinite-dimensional matrices with unbounded eigenvalues, one has to be more careful with this analysis.

**Exercise 3.8.** a) Prove that

$$e^{-i\theta} = \cos\theta - i\sin\theta \tag{3.113}$$

*without* using the power series expansion. Hint: define the functions

$$\begin{aligned} P(\theta) &= \cos\theta - i\sin\theta, & (3.114)\\ Q(\theta) &= -e^{-i\theta}, & (3.115) \end{aligned}$$

and show that they obey the same first-order differential equation

$$\begin{aligned} \frac{d}{d\theta}P(\theta) &= -iP(\theta), & (3.116)\\ \frac{d}{d\theta}Q(\theta) &= -iQ(\theta) & (3.117) \end{aligned}$$

and have the same initial condition

$$P(0) = Q(0) = 1. \tag{3.118}$$

b) Use the same method to prove that

$$e^{-i\frac{\chi}{2}\sigma^z} = \cos\left(\frac{\chi}{2}\right)\hat{I} - i\sin\left(\frac{\chi}{2}\right)\sigma^z. \tag{3.119}$$

Using the fact that $\sigma^z|0\rangle = (+1)|0\rangle$ and $\sigma^z|1\rangle = (-1)|1\rangle$, yields

$$\begin{aligned} U_z(\chi)|0\rangle &= e^{-i\frac{\chi}{2}}|0\rangle & (3.120)\\ U_z(\chi)|1\rangle &= e^{+i\frac{\chi}{2}}|0\rangle. & (3.121) \end{aligned}$$

Applying this to the standard state $|\hat{n}\rangle$, and using the fact that $\sigma^z|0\rangle = (+1)|0\rangle$ and $\sigma^z|1\rangle = (-1)|1\rangle$, yields

$$\begin{aligned} U_z(\chi)|\hat{n}\rangle &= U_z(\chi)\left[\cos\frac{\theta}{2}|0\rangle + \sin\frac{\theta}{2}e^{i\varphi}|1\rangle\right] & (3.122)\\ &= \left[\cos\frac{\theta}{2}e^{-i\frac{\chi}{2}}|0\rangle + \sin\frac{\theta}{2}e^{i(\varphi+\frac{\chi}{2})}|1\rangle\right] & (3.123)\\ &= e^{-i\frac{\chi}{2}}\left[\cos\frac{\theta}{2}|0\rangle + \sin\frac{\theta}{2}e^{i(\varphi+\chi)}|1\rangle\right] & (3.124)\\ &= e^{-i\frac{\chi}{2}}|R_z(\chi)\hat{n}\rangle & (3.125)\\ &= e^{-i\frac{\chi}{2}}|\hat{n}'\rangle. & (3.126) \end{aligned}$$

Thus the $2 \times 2$ complex matrix $U_z(\chi)$ correctly rotates the quantum state by an angle $\chi$ around the $z$ axis of the Bloch sphere. Notice however that the resulting state differs from the standard state by an (irrelevant) global phase factor. This is because we made the arbitrary choice to have the standard state parametrization for $|\hat{n}\rangle$ yield a coefficient of $|0\rangle$ that is purely real so that the only complex amplitude is found in the coefficient of $|1\rangle$.

Notice that $U_z(-\chi) = U_z^{-1}(\chi) = U_z^\dagger(\chi)$. Hence $U_z(\chi)$ is *unitary*, meaning that

$$U_z^\dagger U_z = \hat{I}. \tag{3.127}$$

All rotations in Hilbert space are unitary and, as discussed earlier, it is straightforward to show that any unitary transformation $U$ preserves the inner product in the Hilbert space. Thus unitary matrices for complex vector spaces are the analog of orthogonal matrices for real vector spaces. The inverse of an orthogonal matrix is its transpose

$$\mathcal{O}^{-1} = \mathcal{O}^{\mathrm{T}}. \tag{3.128}$$

The inverse of a unitary matrix is its conjugate transpose

$$U^{-1} = U^\dagger = \left(U^{\mathrm{T}}\right)^*, \tag{3.129}$$

which is the natural generalization of orthogonal rotation matrices for vector spaces over the reals to complex vector spaces.

**Exercise 3.9.** Prove that the Pauli matrices $\sigma^x, \sigma^y, \sigma^z$ are both Hermitian and unitary. This fact can be confusing for beginning students. Because the Pauli matrices are Hermitian they correspond to physical observables that can be measured. Because they are unitary they also correspond to rotation operations that can be applied to quantum states. These are two very different things.

**Exercise 3.10.**
a) Prove that every operator that is both Hermitian and unitary squares to the identity. Using this, prove that if such operators are traceless, their spectrum contains only $+1$ and $-1$ and has equal numbers of each.

b) Conversely, prove the following lemma: Every operator that both squares to the identity and is Hermitian, is necessarily unitary.

**Exercise 3.11.** Every $N \times N$ unitary matrix can be written in the form

$$U = e^{i\theta\hat{M}}, \tag{3.130}$$

where $\theta$ is a real parameter and $\hat{M}$ is an $N \times N$ Hermitian matrix.

a) Using the fact that the eigenvectors of an Hermitian matrix form a complete basis, find the spectrum (set of eigenvalues) and eigenvectors of $U$ in terms of the eigenvalues and eigenvectors in terms of $\hat{M}$.

b) Using this result, prove that $U$ is unitary.

So far we have only considered rotations around the $z$ axis of the Bloch sphere. By analogy with Eq. (3.110), we can rotate the state around the $x$ axis through an angle $\chi$ via the unitary

$$U_x(\chi) = e^{-i\frac{\chi}{2}\sigma^x}, \tag{3.131}$$

around the $y$ axis via

$$U_y(\chi) = e^{-i\frac{\chi}{2}\sigma^y}, \tag{3.132}$$

and around an arbitrary $\hat{\omega}$ axis via

$$U_{\hat{\omega}}(\chi) = e^{-i\frac{\chi}{2}\hat{\omega}\cdot\vec{\sigma}}, \tag{3.133}$$

where

$$\hat{\omega}\cdot\vec{\sigma} = \omega_x\sigma^x + \omega_y\sigma^z + \omega_z\sigma^z \tag{3.134}$$

is a linear combination of the three Pauli matrices.[5]

Mathematically, the three components of the qubit spin polarization $\vec{S} = \frac{1}{2}(\sigma^x, \sigma^y, \sigma^z)$ are the generators of (the Lie group of) rotations in spin space (i.e., rotations on the Bloch sphere). As an example, let us consider a rotation by angle $\pi/2$ around the $y$ axis to see how the cardinal points on the Bloch

[5]It is important to note that since the Pauli matrices do not commute with each other, the exponential of the sum of Pauli matrices cannot be written as a simple product of the individual exponentials.

sphere transform into each other as expected:

$$\begin{aligned}
U_y(\frac{\pi}{2})|+Z\rangle &= \left[\cos\frac{\pi}{4}\hat{I} - i\sin\frac{\pi}{4}\sigma^y\right]|0\rangle = \frac{1}{\sqrt{2}}\left[|0\rangle + |1\rangle\right] \\
&= +|+X\rangle, && (3.135)\\
U_y(\frac{\pi}{2})|-Z\rangle &= -|-X\rangle && (3.136)\\
U_y(\frac{\pi}{2})|+X\rangle &= +|-Z\rangle && (3.137)\\
U_y(\frac{\pi}{2})|-X\rangle &= +|+Z\rangle && (3.138)
\end{aligned}$$

where we have used the fact that

$$\begin{aligned}
-i\sigma^y|0\rangle &= -i\begin{pmatrix} 0 & -i \\ +i & 0 \end{pmatrix}\begin{pmatrix} 1 \\ 0 \end{pmatrix} = \begin{pmatrix} 0 \\ 1 \end{pmatrix} = |1\rangle && (3.139)\\
-i\sigma^y|1\rangle &= -i\begin{pmatrix} 0 & -i \\ +i & 0 \end{pmatrix}\begin{pmatrix} 0 \\ 1 \end{pmatrix} = -\begin{pmatrix} 1 \\ 0 \end{pmatrix} = -|0\rangle && (3.140)
\end{aligned}$$

**Exercise 3.12.** Prove that

$$\begin{aligned}
U_x(\frac{\pi}{2})|+Z\rangle &= |-Y\rangle && (3.141)\\
U_x(\frac{\pi}{2})|-Z\rangle &= -i|+Y\rangle. && (3.142)
\end{aligned}$$

SMG NEEDS WORK: This discussion needs to be improved. Most general unitary requires three angles like Euler. Finding the unitary that takes $|0\rangle$ to an arbitrary state is not the most general unitary!

Readers familiar with classical mechanics may be aware that to rotate an ordinary physical object in 3 dimensions from an initial configuration to an arbitrary final configuration (generically) requires specification of 3 so-called Euler angles. For single qubit quantum states, we need only specify two rotations to rotate the initial state $|0\rangle$ to an arbitrary final state $|+\hat{n}\rangle$. We could for example rotate by angle $\theta$ around the $y$ axis and then by angle $\phi$

around the $z$ axis. This would yield

$$\begin{aligned}
U|0\rangle &= e^{-i\frac{\varphi}{2}Z}e^{-i\frac{\theta}{2}Y} && (3.143)\\
&= e^{-i\frac{\varphi}{2}Z}\left[\cos\frac{\theta}{2}I - i\sin\frac{\theta}{2}Y\right]|0\rangle && (3.144)\\
&= e^{-i\frac{\varphi}{2}Z}\left[\cos\frac{\theta}{2}|0\rangle + \sin\frac{\theta}{2}|1\rangle\right] && (3.145)\\
&= \left[e^{-i\frac{\varphi}{2}}\cos\frac{\theta}{2}|0\rangle + e^{+i\frac{\varphi}{2}}\sin\frac{\theta}{2}|1\rangle\right] && (3.146)\\
&= e^{-i\frac{\varphi}{2}}\left[\cos\frac{\theta}{2}|0\rangle + e^{+i\varphi}\sin\frac{\theta}{2}|1\rangle\right] && (3.147)\\
&= e^{-i\frac{\varphi}{2}}|+\hat{n}\rangle. && (3.148)
\end{aligned}$$

Thus we obtain the desired final state up to an irrelevant global phase. We could get rid of this phase by a third and final rotation around the $\hat{n}$ axis which would correspond to the 3rd Euler angle in the classical case. This shows us that the phase in question is not important we are only looking at state $|+\hat{n}\rangle$ or $|-\hat{n}\rangle$ since the final rotation just changes the global phase. However if we are looking at a superposition of these two states, the spin vector is point in some direction not co-linear with $\hat{n}$ and the final rotation changes the spin vector since it produces an important *relative* phase between the two states in the superposition.

Some authors choose to define the Z rotation unitary to be

$$U_1(\varphi) = e^{-i\frac{\varphi}{2}[Z-I]} = e^{+i\frac{\varphi}{2}}U_z(\varphi), \tag{3.149}$$

so that the entire relative phase is achieved by changing the phase only of the $|1\rangle$

$$\begin{aligned}
U_1(\varphi)|0\rangle &= |0\rangle, && (3.150)\\
U_1(\varphi)|1\rangle &= e^{i\varphi}|1\rangle && (3.151)
\end{aligned}$$

to be consistent with the phase choice made in the definition of $|+\hat{n}\rangle$ and we have

$$U_1(\varphi)e^{-i\frac{\theta}{2}Y}|0\rangle = |+\hat{n}\rangle. \tag{3.152}$$

### 3.4.1 The Hadamard Gate

We see from Eqs. (3.135-3.138) that a rotation by $\pi/2$ around the $y$ axis interchanges the states $\pm Z$ and $\pm X$, but not in a simple way. In order to swap the $x$ and $z$ components of the polarization vector without the inconvenience of the various minus signs in Eqs. (3.136-3.138), quantum computer scientists like to invoke the Hadamard gate

$$H = |+X\rangle\langle +Z| + |-X\rangle\langle -Z| \tag{3.153}$$

$$= \frac{1}{\sqrt{2}}(\sigma^x + \sigma^z) = \frac{1}{\sqrt{2}}\begin{pmatrix} 1 & 1 \\ 1 & -1 \end{pmatrix} \tag{3.154}$$

which has the following nice properties

$$H|0\rangle = |+\rangle \tag{3.155}$$

$$H|1\rangle = |-\rangle \tag{3.156}$$

$$H|+\rangle = |0\rangle \tag{3.157}$$

$$H|-\rangle = |1\rangle. \tag{3.158}$$

**Exercise 3.13.** Using Eq. (3.153), prove that the Hadamard gate obeys

a) $H = |+Z\rangle\langle +X| + |-Z\rangle\langle -X|$,

b) $H^2 = I$,

c) $H$ is unitary.

d) $H$ is a rotation of the general form given in Eq. (3.133). Find the parameters $\chi$ and $\hat{\omega}$.

## 3.5 Hilbert Space and Operators for Multiple Qubits

So far we have been focused on the two-dimensional complex vector space describing the quantum states of a single spin or qubit. We turn now to the study of how to extend this to two qubits and ultimately to $N$ qubits. If we have a single qubit, the standard basis has two states, $|0\rangle$ and $|1\rangle$ and the qubit can be in any linear superposition of those two basis states. If we have two qubits, we need four basis states: $|11\rangle, |10\rangle, |0,1\rangle, |00\rangle$. We see that the states are labeled by the binary numbers $00, 01, 10, 11$ corresponding to the base-10 numbers $0, 1, 2, 3$ just as in a classical computer memory that contains only two bits. Our quantum bits can however be in a superposition of all four states of the form

$$|\psi\rangle = \psi_{11}|11\rangle + \psi_{10}|10\rangle + \psi_{01}|01\rangle + \psi_{00}|00\rangle, \tag{3.159}$$

or equivalently a column vector of length four

$$|\psi\rangle = \begin{pmatrix} \psi_{00} \\ \psi_{01} \\ \psi_{10} \\ \psi_{11} \end{pmatrix}. \tag{3.160}$$

Note that the choice of how to order the entries in the column is completely arbitrary, but once you make a choice you must stick to it for all your calculations. For the case of $N$ qubits, all of the above generalizes to a vector space of dimension $2^N$. We shall always use the standard ordering which lists the entries in the column according to the binary numbering order used above.

We have so far seen two kinds of products for vectors, the inner product $\langle\psi|\phi\rangle$ which is a scalar (complex number), and the outer product $|\phi\rangle\langle\psi|$ which is an operator that has a matrix representation. When it comes to thinking about the quantum states of a composite physical system consisting of multiple qubits, we have to deal with yet another kind of product, the *tensor product.* The tensor product of the Hilbert space $\mathcal{H}_1$ of the first qubit with that of a second qubit $\mathcal{H}_2$ yields a new Hilbert space $\mathcal{H}_1 \otimes \mathcal{H}_2$ whose dimension is the product of the dimensions of the two individual Hilbert spaces. The two-qubit basis states in Eq. (3.159) can be thought of as tensor products of individual vectors,

$$|01\rangle = |0\rangle \otimes |1\rangle. \tag{3.161}$$

The tensor product of two column vectors of length $n_1$ and $n_2$ is a column vector of length $n_1 n_2$. For example, the two-qubit states in the standard basis are

$$
\begin{aligned}
|00\rangle &= |0\rangle \otimes |0\rangle = \begin{pmatrix} 1 \\ 0 \end{pmatrix} \otimes \begin{pmatrix} 1 \\ 0 \end{pmatrix} = \begin{pmatrix} 1 \begin{pmatrix} 1 \\ 0 \end{pmatrix} \\ 0 \begin{pmatrix} 1 \\ 0 \end{pmatrix} \end{pmatrix} = \begin{pmatrix} 1 \\ 0 \\ 0 \\ 0 \end{pmatrix} && (3.162) \\
|01\rangle &= |0\rangle \otimes |1\rangle = \begin{pmatrix} 1 \\ 0 \end{pmatrix} \otimes \begin{pmatrix} 0 \\ 1 \end{pmatrix} = \begin{pmatrix} 1 \begin{pmatrix} 0 \\ 1 \end{pmatrix} \\ 0 \begin{pmatrix} 0 \\ 1 \end{pmatrix} \end{pmatrix} = \begin{pmatrix} 0 \\ 1 \\ 0 \\ 0 \end{pmatrix} && (3.163) \\
|10\rangle &= |1\rangle \otimes |0\rangle = \begin{pmatrix} 0 \\ 1 \end{pmatrix} \otimes \begin{pmatrix} 1 \\ 0 \end{pmatrix} = \begin{pmatrix} 0 \begin{pmatrix} 1 \\ 0 \end{pmatrix} \\ 1 \begin{pmatrix} 1 \\ 0 \end{pmatrix} \end{pmatrix} = \begin{pmatrix} 0 \\ 0 \\ 1 \\ 0 \end{pmatrix} && (3.164) \\
|11\rangle &= |1\rangle \otimes |1\rangle = \begin{pmatrix} 0 \\ 1 \end{pmatrix} \otimes \begin{pmatrix} 0 \\ 1 \end{pmatrix} = \begin{pmatrix} 0 \begin{pmatrix} 0 \\ 1 \end{pmatrix} \\ 1 \begin{pmatrix} 0 \\ 1 \end{pmatrix} \end{pmatrix} = \begin{pmatrix} 0 \\ 0 \\ 0 \\ 1 \end{pmatrix} && (3.165)
\end{aligned}
$$

Notice that we compute the tensor product of two vectors by inserting the second vector (i.e., the one on the right) in the product into the first vector as illustrated in the expressions in the fourth column of the array of equations above. [Note: It is crucial that you maintain the correct ordering convention.] For a general pair of two-qubit states, the tensor product is

$$
\begin{pmatrix} v_0 \\ v_1 \end{pmatrix} \otimes \begin{pmatrix} u_0 \\ u_1 \end{pmatrix} = \begin{pmatrix} v_0 \begin{pmatrix} u_0 \\ u_1 \end{pmatrix} \\ v_1 \begin{pmatrix} u_0 \\ u_1 \end{pmatrix} \end{pmatrix} = \begin{pmatrix} v_0u_0 \\ v_0u_1 \\ v_1u_0 \\ v_1u_1 \end{pmatrix}. \tag{3.166}
$$

How do we represent operators in this multi-qubit space? Physicists sometimes put labels on the operators to denote which qubit they are acting on. For example, the $x$ component of the spin vector of the $j$th qubit is denoted $\sigma_j^x$, with $j = 0, 1$. We will find it convenient to label the qubits in an $N$-qubit register starting with 0 for the right most qubit and ending with $N - 1$ for the left most qubit like this

$$
|q_{N-1} \ldots q_2q_1q_0\rangle. \tag{3.167}
$$

Using this convention, we have for example

$$\sigma_0^x|00\rangle = |01\rangle \tag{3.168}$$
$$\sigma_1^x|00\rangle = |10\rangle \tag{3.169}$$
$$\sigma_1^x\sigma_0^x|00\rangle = |11\rangle \tag{3.170}$$
$$\sigma_1^z\sigma_1^x|00\rangle = \sigma_1^z|10\rangle = -|10\rangle. \tag{3.171}$$

More formally however, we need to recognize that the Hilbert space for two qubits is the tensor product of their individual Hilbert spaces and therefore states are represented as column vectors of length 4. This means that all linear operators are represented by $4\times 4$ matrices. Therefore we should write two-qubit operators as the *Kronecker product* (also known confusingly and sloppily as the outer, direct or tensor product) of the $2\times 2$ matrices representing the individual qubit operators. [AUTHOR NOTE: NEED TO DEFINE ALL THESE CONCEPTS EXPLICITLY AND ACCURATELY HERE OR IN APP. B. DO ANY OF YOU MATH MAJORS HAVE SUGGESTIONS?] The Kronecker product of a matrix with dimension $n_1\times n_1$ with a matrix with dimension $n_2\times n_2$ is a larger matrix of dimension $(n_1n_2)\times(n_1n_2)$. This is of course consistent with the fact that the dimension of the Kronecker product Hilbert space is $n_1n_2$. As an example of a Kronecker product consider the matrix representation of the operator $\sigma_1^z$

$$\sigma^z\otimes\sigma^0 = \begin{pmatrix} +1 & 0 \\ 0 & -1 \end{pmatrix} \otimes \begin{pmatrix} +1 & 0 \\ 0 & +1 \end{pmatrix}. \tag{3.172}$$

This is the formal way of representing $\sigma_1^z$ that acts on qubit 1 with the Pauli $Z$ operator and does nothing to the 0th qubit (applies the identity). Notice that we have removed the qubit labels because the position of the operator in the Kronecker product implies which qubit it acts on. The rule for how you write out the entries to this $4\times 4$ matrix depends on exactly how you order the terms in the column vector in Eq. (3.160). It is clear however that we want the following to be true (assuming we number the qubits in $|01\rangle$ from right to left inside the Dirac ket)

$$\sigma^z\otimes\sigma^0|00\rangle = (+1)|00\rangle \tag{3.173}$$
$$\sigma^z\otimes\sigma^0|01\rangle = (+1)|01\rangle \tag{3.174}$$
$$\sigma^z\otimes\sigma^0|10\rangle = (-1)|10\rangle \tag{3.175}$$
$$\sigma^z\otimes\sigma^0|11\rangle = (-1)|11\rangle, \tag{3.176}$$

and for the operator $\sigma_0^z$ we have

$$\begin{aligned}
\sigma^0 \otimes \sigma^z|00\rangle &= (+1)|00\rangle && (3.177)\\
\sigma^0 \otimes \sigma^z|01\rangle &= (-1)|01\rangle && (3.178)\\
\sigma^0 \otimes \sigma^z|10\rangle &= (+1)|10\rangle && (3.179)\\
\sigma^0 \otimes \sigma^z|11\rangle &= (-1)|11\rangle. && (3.180)
\end{aligned}$$

Using the ordering of the basis states in the column vector representation in Eq. (3.160), the outer product in Eq. (3.172) is represented by the matrix

$$\sigma^z \otimes \sigma^0 = \begin{pmatrix} +1 & 0 & 0 & 0 \\ 0 & +1 & 0 & 0 \\ 0 & 0 & -1 & 0 \\ 0 & 0 & 0 & -1 \end{pmatrix}. \tag{3.181}$$

As a convenient mnemonic, think of the Kronecker product as producing a $2 \times 2$ array whose entries are the second matrix in the product (the $2 \times 2$ identity matrices in the above example), each multiplied by one of the four entries in the first matrix in the product (the $2 \times 2$ $\sigma_1^z$ matrix in the example above)

$$\begin{aligned}
\sigma^z \otimes \sigma^0 &= \begin{pmatrix} +1 & 0 \\ 0 & -1 \end{pmatrix} \otimes \begin{pmatrix} +1 & 0 \\ 0 & +1 \end{pmatrix} && (3.182)\\
&= \begin{pmatrix} (+1)\begin{pmatrix} +1 & 0 \\ 0 & +1 \end{pmatrix} & (0)\begin{pmatrix} +1 & 0 \\ 0 & +1 \end{pmatrix} \\ (0)\begin{pmatrix} +1 & 0 \\ 0 & +1 \end{pmatrix} & (-1)\begin{pmatrix} +1 & 0 \\ 0 & +1 \end{pmatrix} \end{pmatrix} && (3.183)\\
&= \begin{pmatrix} +1 & 0 & 0 & 0 \\ 0 & +1 & 0 & 0 \\ 0 & 0 & -1 & 0 \\ 0 & 0 & 0 & -1 \end{pmatrix}, && (3.184)
\end{aligned}$$

in agreement with Eq. (3.181). Thus the procedure for computing the Kronecker product of two matrices is directly analogous to the procedure for computing the tensor product of two state vectors.

Reversing the order of the operators in the product gives

$$\sigma^0 \otimes \sigma^z = \begin{pmatrix} +1 & 0 & 0 & 0 \\ 0 & -1 & 0 & 0 \\ 0 & 0 & +1 & 0 \\ 0 & 0 & 0 & -1 \end{pmatrix}. \tag{3.185}$$

We can think of this as four copies of the $\sigma^z$ matrix each multiplied by the appropriate entry in the identity matrix.

We can also consider the *sum* (Kronecker sum) of two operator–not to be confused with the 'direct sum' which is different). For example, the $z$ component of the total spin vector is given by

$$S^z \equiv \sigma_1^z + \sigma_0^z = \sigma^z \oplus \sigma^z, \tag{3.186}$$

is represented in matrix form by

$$S^z = \sigma^z \otimes \sigma^0 + \sigma^0 \otimes \sigma^z = \begin{pmatrix} +2 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & -2 \end{pmatrix}. \tag{3.187}$$

This matrix form makes sense because it says that

$$\begin{aligned} S^z|00\rangle &= (+2)|00\rangle \\ S^z|01\rangle &= (+0)|01\rangle \\ S^z|10\rangle &= (+0)|10\rangle \\ S^z|11\rangle &= (-2)|11\rangle. \end{aligned}$$

All this notation can be a bit confusing. To summarize: the symbols $\otimes$ and $\oplus$ refer to Kronecker product and Kronecker sum (not the 'direct' sum, which is different)

`https://en.wikipedia.org/wiki/Matrix_addition,`

although confusingly the Kronecker product is sometimes referred to as the matrix direct product,

`https://en.wikipedia.org/wiki/Kronecker_product.`

In Wolfram *Mathematica*®, the following commands will produce Kronecker products of operators

```
II = IdentityMatrix[2];
MatrixForm[II]
Z = {{1, 0}, {0, -1}};
MatrixForm[Z]
ZI = KroneckerProduct[Z, II];
```

```
MatrixForm[ZI]
IZ = KroneckerProduct[II,Z];
MatrixForm[IZ]
MatrixForm[IZ + ZI]
```

and the following commands will produce operators from Kronecker products of their eigenvectors

```
plusystate = {1, I}/Sqrt[2]; minusystate = {1, -I}/Sqrt[2];
Y = MatrixForm[
KroneckerProduct[plusystate, Conjugate[plusystate]] -
KroneckerProduct[minusystate, Conjugate[minusystate]]]
```

The above is equivalent to

$$Y = \frac{1}{2}\begin{pmatrix} 1 \\ +i \end{pmatrix} \begin{pmatrix} 1 & +i \end{pmatrix}^* - \frac{1}{2}\begin{pmatrix} 1 \\ -i \end{pmatrix} \begin{pmatrix} 1 & -i \end{pmatrix}^* \tag{3.188}$$

**Exercise 3.14.** The Kronecker products of operators we have been studying have simple distributive properties. Show that for general qubit states $|\psi\rangle$ and $|\phi\rangle$

$$(\sigma^z \otimes \sigma^x)(|\psi\rangle \otimes |\phi\rangle) = (\sigma^z |\psi\rangle) \otimes (\sigma^x |\phi\rangle).$$

Hint: Do this using the matrix and column vector representations of each of the terms on each side of the equation.

**Exercise 3.15.** Write out the matrix representation of the following two-qubit Pauli matrices

a) $\sigma_1^x \sigma_0^x = \sigma^x \otimes \sigma^x$, where the subscripts 0 and 1 refer to which qubit the operator acts upon.

b) $\sigma_1^x + \sigma_0^x = \sigma^x \oplus \sigma^x = (\sigma^x \otimes \hat{I}) + (\hat{I} \otimes \sigma^x)$.

c)

$$\begin{aligned} (\sigma_1^0 + \sigma_1^z) + \frac{1}{2}(\sigma_0^0 + \sigma_0^z) &= \left(\sigma^0 + \sigma^z\right) \oplus \frac{1}{2}\left(\sigma^0 + \sigma^z\right) \\ &= \left[\left(\sigma^0 + \sigma^z\right) \otimes \hat{I}\right] + \frac{1}{2}\left[\hat{I} \otimes \left(\sigma^0 + \sigma^z\right)\right]. \end{aligned}$$

**Box 3.4. The No-cloning Theorem** The no-cloning theorem [1] states that it is impossible to make a copy of an unknown quantum state.

The essential idea of the no-cloning theorem is that in order to make a copy of an unknown quantum state, you would have to measure it to see what the state is and then use that knowledge to make the copy. However measurement of the state produces in general a random measurement result and random back action (state collapse) and thus it is not possible to fully determine the state. This is a reflection of the fact that measurement of a qubit yields one classical bit of information which is not enough in general to fully specify the state via its co-latitude $\theta$ and longitude $\varphi$ on the Bloch sphere.

Of course if you have prior knowledge, such as the fact that the state is an eigenstate of $\sigma^x$, then a measurement of $\sigma^x$ tells you the $\pm 1$ eigenvalue and hence the $|\pm X\rangle$ state. The measurement gives you one additional classical bit of information which is all you need to have complete knowledge of the state.

A more formal statement of the no-cloning theorem is the following. Given an unknown state $|\psi\rangle = \alpha|0\rangle + \beta|1\rangle$ and an ancilla qubit initially prepared in a definite state (e.g. $|1\rangle$), there does not exist a unitary operation $U$ that will take the initial joint state

$$|\Phi\rangle = [\alpha|0\rangle + \beta|1\rangle] \otimes |1\rangle \tag{3.189}$$

to the final state

$$\begin{aligned} U|\Phi\rangle &= [\alpha|0\rangle + \beta|1\rangle] \otimes [\alpha|0\rangle + \beta|1\rangle], \\ &= \alpha^2|00\rangle + \alpha\beta[|01\rangle + |10\rangle] + \beta^2|11\rangle. \end{aligned} \tag{3.190}$$

unless $U$ depends on $\alpha$ and $\beta$.

The proof is straightforward. The RHS of Eq. (3.189) is linear in $\alpha$ and $\beta$, whereas the RHS of Eq. (3.190) is quadratic. This is impossible unless $U$ depends on $\alpha$ and $\beta$. For an unknown state, we do not know $\alpha$ and $\beta$ and therefore cannot construct $U$.

**Exercise 3.16.** Assuming you have knowledge of $\alpha$ and $\beta$, construct an explicit unitary $U(\alpha, \beta)$ that carries out the cloning operation in Eq. (3.190).

**Exercise 3.17.** Bob attempts to clone an unknown state $|\psi\rangle = |\hat{n}\rangle$ that Alice has given him. Its orientation $\hat{n}$ could be anywhere on the Bloch sphere with equal a priori probability. Bob chooses to measure the qubit along an arbitrary measurement axis $\hat{m}$ by measuring $M = \hat{m} \cdot \vec{\sigma}$. Based on the measurement result Bob guesses that the qubit was in state $| + \hat{m}\rangle$ (if the measurement result was $+1$) or state $| - \hat{m}\rangle$ (if the measurement result was $-1$). Knowing this result Bob can then make an arbitrary number of approximate clones. What is the average fidelity of this approximate cloning process? Hint: WLOG you can take $\hat{m} = (0, 0, 1)$.

The reader is reminder that a handy summary of the key equations in this chapter is provided in App. C.

# Chapter 4

# Two-Qubit Gates and Multi-Qubit Entanglement

## 4.1 Introduction: Multi-Qubit Operations and Measurements

We learned in Chap. 3 that the physical operations we are allowed to carry out (besides measurement) are represented by unitary matrices acting on state vectors. These operations are rotations in Hilbert space and transform one state into another. Indeed, this is all a quantum computer can do. You load a quantum state into the $N$ qubits of the input register, carry out a unitary rotation in the $2^N$-dimensional Hilbert space, and place the resulting state into the $N$ qubits of the output register. Measure the state of the output register and you acquire $N$ classical bits of information.[1] Unitary operations can be combined with measurements in interesting ways. After carrying out a unitary $U_0$, one could measure a subset of say $m$ qubits. The measurement will yield $m$ classical bits of information corresponding to one instance out of $2^m$ possible results. Conditioned on this, one can choose which

[1]As noted earlier, $N$ classical bits is far less information than it takes to specify a generic $N$-qubit state. Thus generically the measurements give random results as the measured state collapses. However certain algorithms, e.g. Shor's factoring algorithm use quantum interference to 'focus' the amplitudes in the output state that it consists of a superposition of only a small number of states in the standard basis and thus the result is not very random. Repeating the algorithm a small number of times yields the desired prime factors with high probability.

of $2^m$ subsequenty unitaries $U_j$ from the set $U_1, U_2, \dots, U_{2^m}$ to apply. The net effect of these three steps is the most general possible quantum operation and is *not* equivalent to a unitary transformation. Such quantum operations can be more powerful for certain purposes (e.g. quantum error correction).

## 4.2 Multi-Qubit Measurements

We have already studied measurements on a single qubit where the Born rule tells us that a measurement of $\hat{M} = \vec{\sigma} \cdot \hat{n} = (|+\hat{n}\rangle\langle +\hat{n}|) - (|-\hat{n}\rangle\langle -\hat{n}|)$ (i.e., made in the $|\pm\hat{n}\rangle$ basis) on state

$$|\psi\rangle = \alpha|+\hat{n}\rangle + \beta|-\hat{n}\rangle, \tag{4.1}$$

yields as a measurement result the $M = 1$ eigenvalue of $\hat{M}$ with probability $|\alpha|^2$ and the $M = -1$ eigenvalue with probability $|\beta|^2$. Recall that in order to apply the Born rule to determine the measurement probabilities, it is essential to re-express the given state in the basis of eigenvectors of the operator being measured. The state collapses to the eigenvector corresponding to the measured eigenvalue.

Multi-qubit measurements follow the same rule. To apply the Born rule you must express the state in the eigenbasis of the multi-qubit operator being measured. For simplicity, we will focus on measurements of operators that are diagonal in the standard basis, but keep in mind that if you are measuring non-diagonal operators, you need to re-express the multi-qubit state in terms of the eigenvectors of the operator being measured.

As preparation for the discussion below, the reader is encouraged to review the discussion of joint probability distributions of multiple variables in Sec. A.4 in Appendix A. For simplicity let us consider a generic two-qubit state in the standard basis

$$|\psi\rangle = \alpha_{00}|00\rangle + \alpha_{01}|01\rangle + \alpha_{10}|10\rangle + \alpha_{11}|11\rangle. \tag{4.2}$$

Measurement of the state of each of the two qubits yields result $(x, y)$ (where $x \in \{0, 1\}$ and $y \in \{0, 1\}$) with Born-rule probability $P(x, y) = |\alpha_{xy}|^2$ and the two-qubit state correspondingly collapses to $|xy\rangle$.

Because we have more than one qubit, we can now ask a new question: What happens if we only measure one of the qubits but not the other? Suppose we only measure $q_0$ (qubit 0, numbering the qubits from right to left as usual). How does the state collapse? To answer this, let us rewrite Eq. (4.2) in the following form which has the state of qubit 0 factored out on the right:

$$|\psi\rangle = \big[\alpha_{00}|0\rangle + \alpha_{10}|1\rangle\big]|0\rangle + \big[\alpha_{01}|0\rangle + \alpha_{11}|1\rangle\big]|1\rangle. \tag{4.3}$$

Let us measure $q_0$ in the standard basis using the operator $\hat{B}_0 = \hat{I} \otimes |1\rangle\langle 1|$. This operator has a doubly degenerate eigenvalue of 0 with eigenvectors of the form

$$|\phi_0\rangle = |\mu\rangle|0\rangle, \tag{4.4}$$

where $|\mu\rangle$, the state of $q_1$, is arbitrary ($|0\rangle$, $|1\rangle$ or any linear combination). The operator also has a doubly degenerate eigenvalue of 1 with eigenvectors of the form

$$|\phi_0\rangle = |\nu\rangle|1\rangle, \tag{4.5}$$

where $|\nu\rangle$ is arbitrary.

Hence if we obtain the measurement result 1, the two-qubit state has to collapse to a new state consistent with that result, namely

$$|\psi_1\rangle = \frac{1}{\sqrt{P_0(1)}}\big[\alpha_{01}|0\rangle + \alpha_{11}|1\rangle\big]|1\rangle, \tag{4.6}$$

where the square root factor produces the correct normalization for the new state and

$$P_0(1) = |\alpha_{01}|^2 + |\alpha_{11}|^2 = P(0,1) + P(1,1) \tag{4.7}$$

is the Born rule probability for obtaining measurement result 1 on qubit 0. The sum reflects the fact that there are two independent ways to obtain this measurement result and their probabilities add. (We do not add the probability amplitudes before squaring because the two states are orthogonal rather than identical.)

Similarly if we obtain the measurement result 0, the two-qubit state collapses to

$$|\psi_0\rangle = \frac{1}{\sqrt{P_0(0)}}\big[\alpha_{00}|0\rangle + \alpha_{10}|1\rangle\big]|0\rangle, \tag{4.8}$$

where

$$P_0(0) = |\alpha_{00}|^2 + |\alpha_{10}|^2. \tag{4.9}$$

In both cases, we see that measurement of $q_0$ produces only a 'partial collapse' of the state that leaves $q_1$ still in a superposition state (dependent on the measurement outcome). This concept of partial collapse will be very important when we study Simon's algorithm and also when we study quantum error correction. The idea behind quantum error correction is that we can make a measurement that that collapses a multi-qubit state enough to tell us what error occurred but does not collapse the state enough to destroy the quantum information being stored in that multi-qubit state, thereby allowing us to correct the error without learning anything about the state in which the error occurred! Much more on this later.

Having seen the result of measuring $q_0$ and finding that $q_1$ has been left in a superposition state, we can now investigate what happens when we subsequently measure $q_1$ (and thus complete the full collapse of the state). Suppose that the measurement of $q_0$ yielded the result 1 so that the partially collapsed state is given by Eq. 4.6. From this state we deduce that a measurement of $q_1$ conditioned on the measurement of $q_0$ having previously yielded 1, obtains result 0 with probability

$$P(0|1) = \frac{1}{P_0(1)}|\alpha_{01}|^2. \tag{4.10}$$

Thus the overall probability $P(0,1)$ to obtain 1 for $q_0$ and 0 for $q_1$ is $P_0(1)$, the probability to obtain 1 for the $q_0$ measurement, times the conditional probability $P(0|1)$ for obtaining 0 for measurement of $q_1$ given that we previously obtained 1 for $q_0$:

$$P(0,1) = P(0|1)P_0(1) = |\alpha_{01}|^2. \tag{4.11}$$

This of course is exactly the Born-rule probability we obtained for the result $(0,1)$ from simultaneous measurement of these two commuting observables.

One can summarize all this more formally in the following way. Suppose that we have a n $n$-qubit system in state $|\psi\rangle$ and we measure a single qubit $q_j$ to be in state $|b_j\rangle$. Then the state of the system (partially) collapses to

$$|\psi'\rangle = \frac{\hat{P}_j(b_j)|\psi\rangle}{\langle\psi|\hat{P}_j(b_j)|\psi\rangle^{1/2}}, \tag{4.12}$$

where the projector is onto the subspace of the Hilbert space that is consistent with the measurement result (as opposed to projecting onto the single state consistent with the measurement result as occurs with a single qubit):

$$\hat{P}_j(b_j) = \hat{I}_n \hat{I}_{n-1} \dots \hat{I}_{j+1} \left[ |b_j\rangle\langle b_j| \right] \hat{I}_{j-1} \dots \hat{I}_1 \hat{I}_0. \tag{4.13}$$

A more subtle case is the one in which we measure a two-qubit operator such as $Z_1 Z_0$. A measurement result of $+1$ only tells us that the two qubits are in the same state, but not which state they are in. Thus the corresponding projector for (just) those two qubits is

$$\hat{P}_{10}(+1) = |00\rangle\langle 00| + |11\rangle\langle 11|. \tag{4.14}$$

Similarly if the measurement result is $-1$ the two qubits must be in opposite states and the corresponding projector is

$$\hat{P}_{10}(-1) = |01\rangle\langle 01| + |10\rangle\langle 10|. \tag{4.15}$$

This type of multi-qubit joint measurement will play a crucial role in quantum error correction and in entanglement generation via measurement. Note the important fact that joint measurement of $Z_1 Z_0$ is completely different than the product of separate measurement results $Z_1$ and $Z_0$. The joint measurement yields one bit of classical information, while with separate measurements you learn two classical bits of information, namely the individual values for each qubit. Thus the state collapse is more complete in the latter case.

How do we actually make a joint measurement of an operator like $Z_1 Z_0$ without learning about the individual qubit states? This is subtle and understanding how to design a circuit for making joint measurements requires us to first learn how to execute two-qubit gates. We turn to this topic in the next section.

## 4.3 Two-Qubit Gates

Let us concentrate for now on two-qubit gates (unitary operations). We saw in Chap. 3 that Hilbert space consisting of the direct product of the Hilbert space for two single qubits is four-dimensional and that operators on that space are represented by $4 \times 4$ matrices. It does not seem that stepping up from one qubit to two qubits is going to lead to anything dramatically new.

However, it does–we will explore the fascinating concept of entanglement. Entanglement was deeply disturbing to the founders of quantum mechanics, especially to Einstein who understood and pointed out its spooky implications. The power of entanglement as a quantum resource has come to be appreciated in recent decades and underlies certain key protocols for quantum communication and for quantum computation.

### 4.3.1 The CNOT Gate

The NOT gate in a classical computer simply flips the bit taking 0 to 1 and 1 to 0. The so-called 'controlled-NOT' or CNOT gate allows one to do 'if-then-else' logic with a computer. It is a two-bit gate that flips the state of the target bit conditioned on the state of the control qubit. It has the 'truth table' shown in Table 4.1 from which we see that the target bit is flipped, if and only if, the control bit is in the 1 state.

| CNOT Truth Table | | | |
|---|---|---|---|
| Control In | Target In | Control Out | Target Out |
| 0 | 0 | 0 | 0 |
| 0 | 1 | 0 | 1 |
| 1 | 0 | 1 | 1 |
| 1 | 1 | 1 | 0 |

Table 4.1: Action of the two-bit CNOT gate in the standard basis.

In the classical context one can imagine measuring the state of the control bit and then using that information to control the flipping of the target bit. However in the quantum context, it is crucially important to emphasize that measuring the control qubit would collapse its state. We must therefore avoid any measurements and seek a *unitary* gate which works correctly when the control qubit is in $|0\rangle$ and $|1\rangle$ and even when it is in a superposition of both possibilities $\alpha|0\rangle + \beta|1\rangle$. As we will soon see, it is this latter situation which will allow us to generate entanglement. When the control qubit is in a superposition state, the CNOT gate causes the target qubit to be both flipped and not flipped in a manner that is correlated with the state of the control qubit. As we will see, these are not ordinary classical statistical correlations (e.g. clouds are correlated with rain), but rather special (and powerful) quantum correlations resulting from entanglement.

**Box 4.1. CNOT without Measurement** How can we possibly flip the state of the target qubit conditioned on the state of the control qubit without measuring the control and hence collapsing its state? We know that in many systems we can cause a transition between two quantum levels separated in energy by an amount $\hbar\omega$ by applying an electromagnetic wave of frequency $\omega$ for a certain precise period of time. [In quantum mechanics energy $E = hf = \hbar\omega$ and frequency $f = \frac{\omega}{2\pi}$ are related with a proportionality given by Planck's constant $h = 2\pi\hbar$.] The way to make this bit flip of the target be conditioned on the state of the control is to have an interaction between the two qubits that causes the transition energy of the target depend on the state of the control. For example, consider the Hamiltonian (the energy operator)

$$H = -\left\{\frac{\hbar\omega_0}{2}Z_0 + \frac{\hbar\omega_1}{2}Z_1 + \hbar g Z_1 Z_0\right\}. \tag{4.16}$$

The energy change to flip qubit 0 from $|0\rangle$ to $|1\rangle$ is $\Delta E = \hbar(\omega_0 - gZ_1)$. Thus if we shine light (or microwaves as appropriate) of frequency $\hbar(\omega_0 + g)$, qubit 0 will flip only when qubit 1 is in $|1\rangle$ because that matches the transition frequency. However if qubit 1 is in the state $|0\rangle$, the transition frequency for qubit 0 is shifted to $\hbar(\omega_0 - g)$ and the light has the wrong frequency to cause the transition. See Fig. 4.1 for an illustration of the level scheme.

If the transition energies (frequencies) are unique then we can apply a driving field (electric or magnetic or whatever couples to the qubit) to cause the desired transition and no others. This allows us to flip the target qubit if and only if the control is in a particular state we choose (1 for CNOT, 0 for $\bar{\text{C}}$NOT).

It may seem intuitive that a qubit can be excited by bathing it in photons. It is less intuitive that the qubit can be de-excited the same way. This is called "stimulated emission." The excited qubit can de-excite by emitting a photon. This can happen spontaneously (which is a form of decoherence that can cause errors in quantum computers), but it can be made to happen much faster (and coherently so it is not an error) if the emitted photon joins many other identical photons that are present because of the drive. This effect was predicted by Einstein (very early before the quantum theory was even fully developed) and is the mechanism behind the operation of lasers.

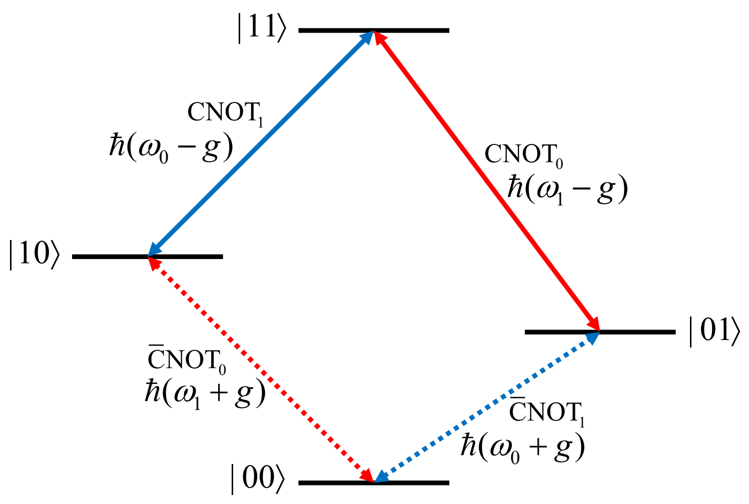

Figure 4.1: Energy levels for two interacting qubits in the computational basis $|q_1 q_0\rangle$ whose Hamiltonian is given by Eq. (4.16). Blue lines correspond to transitions in which qubit $q_0$ is flipped. Red lines correspond to transitions in which qubit $q_1$ is flipped. By choosing $\omega_0, \omega_1$ and $g$ appropriately, all four single qubit transition energies can be made unique, thereby allowing flip of one qubit conditioned on the state of the other. Each transition is labeled by its energy and the corresponding operation CNOT$_j$ where $j$ labels which qubit is the control. Dashed lines correspond to $\bar{\text{C}}$NOT gates in which the control must be in state 0 (rather than 1) in order to flip the target. Altogether there are four different transistions corresponding to four distinct CNOT or $\bar{\text{C}}$NOT gates. If $g = 0$, then the two qubits are not interacting and the transition energy for one qubit is not dependent on the state of the other, so that conditional operations are not possible.

From the classical truth table we can attempt to construct the appropriate quantum operator by putting the dual of initial state ket in the bra and the desired final state in the ket. Number the qubits from right to left (beginning with zero) and letting qubit 0 be the target and qubit 1 be the control, we have

$$\text{CNOT}_1 = |00\rangle\langle 00| + |01\rangle\langle 01| + |11\rangle\langle 10| + |10\rangle\langle 11|, \tag{4.17}$$

where the subscript on CNOT tells us which qubit is the control bit. We see from the orthonormality of the basis states that this produces all the desired transformations, for example

$$\text{CNOT}_1|11\rangle = |10\rangle. \tag{4.18}$$

This however is not enough. We have to prove that the desired transformations are legal. That is, we must show that CNOT is unitary. It is clear by inspection that CNOT is Hermitian. A straightforward calculation shows that $(\mathrm{CNOT})^2 = \hat{I}$. Hence by the lemma in Ex. 3.10b, CNOT is unitary.

It is also instructive to write the gate in the following manner

$$\mathrm{CNOT}_1 = \frac{1}{2}\left(\sigma^0 - \sigma^z\right) \otimes \sigma^x + \frac{1}{2}\left(\sigma^0 + \sigma^z\right) \otimes \sigma^0. \tag{4.19}$$

The first parentheses (including the factor of 1/2) is the projector $|1\rangle\langle 1|$ onto the $|1\rangle$ state for the control qubit (qubit 1). Similarly, the second parentheses (including the factor of 1/2) is the projector $|0\rangle\langle 0|$ onto the $|0\rangle$ state for the control qubit. Thus if the control qubit is in $|1\rangle$, then $\sigma^x$ flips the target qubit while the remaining term vanishes. Conversely when the control qubit is in $|0\rangle$, the coefficient of $\sigma^x$ vanishes and only the identity in the second term acts.

In the standard two-qubit basis defined in Eq. (3.160), the operator in Eq. (4.17) and Eq. (4.19) has the matrix representation

$$\mathrm{CNOT}_1 = \begin{pmatrix} 0 & 0 \\ 0 & 1 \end{pmatrix} \otimes \begin{pmatrix} 0 & 1 \\ 1 & 0 \end{pmatrix} + \begin{pmatrix} 1 & 0 \\ 0 & 0 \end{pmatrix} \otimes \begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix} \tag{4.20}$$

$$= \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 1 & 0 \end{pmatrix}. \tag{4.21}$$

The $\mathrm{CNOT}_1$ unitary is represented in 'quantum circuit' notation by the construction illustrated in left panel of Fig. 4.2.

**Exercise 4.1.** By analogy with Eq. (4.21), find the matrix representation of the $\mathrm{CNOT}_0$ gate given in the right panel of Fig. 4.2

**Exercise 4.2.** Consider the reset operator $R$ defined in Box 4.2. Find a state whose norm is not preserved under $R$. This is further proof that $R$ is not a legal unitary operator and requires the assistance of a measurement operation.

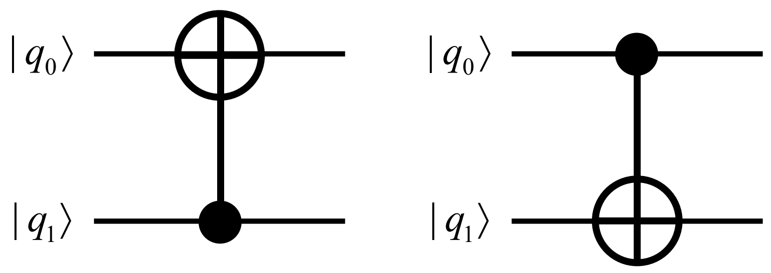

Figure 4.2: Quantum circuit representation of the CNOT$_1$ operation (left panel) and the CNOT$_0$ operation (right panel). The filled circle denotes the control qubit (in the left panel, $q_1$) and the symbol $\oplus$ denotes the target qubit (in the left panel, $q_0$) for the gate. The right panel shows the gate with control and target interchanged. In quantum circuit notation the order in which gates are applied ('time') runs from left to right. If the circuit has GATE$_1$ followed by GATE$_2$, reading from left to right, this corresponds to the (right-to-left) sequence of matrix operations GATE$_2$ GATE$_1|$INPUT STATE$\rangle$. For the $\bar{\text{C}}$NOT gate the control is shown as an open, rather than filled, circle. This denotes the operation being activated by the control being in 0 rather than 1.

**Box 4.2. RESET: Some desired operations are not unitary.** One important task in a quantum computer is to reset all the bits to some standard state before starting a new computation. Let us take the standard state to be $|0\rangle$. We want to map every initial state to $|0\rangle$ which can be done with

$$R = |0\rangle\langle 0| + |0\rangle\langle 1| = \begin{pmatrix} 1 & 1 \\ 0 & 0 \end{pmatrix}. \tag{4.22}$$

$R$ cannot be unitary because a unitary operation preserves inner products between states. The two initial states $|0\rangle$ and $|1\rangle$ are orthogonal and yet they both map to the state $|0\rangle$ which means that the states under the mapping are no longer orthogonal. Equivalently, we see that $R$ is not invertible because we don't know if $R^{-1}$ is supposed to take $|0\rangle$ to $|0\rangle$ or to $|1\rangle$. One can also mechanically check that $R$ is not unitary by writing

$$R^\dagger = |0\rangle\langle 0| + |1\rangle\langle 0|, \tag{4.23}$$

$$R^\dagger R = (|0\rangle\langle 0| + |1\rangle\langle 0|)\,(|0\rangle\langle 0| + |0\rangle\langle 1|) \tag{4.24}$$

$$= \begin{pmatrix} 1 & 1 \\ 1 & 1 \end{pmatrix} = I + \sigma^x \neq I. \tag{4.25}$$

Thus if we only have access to unitary rotations, we cannot reset an unknown state. There is however a simple fix for this problem. If we measure $\sigma^z$ (say), the unknown state will randomly collapse to either $|0\rangle$ or $|1\rangle$. If it collapses to $|0\rangle$, apply the identity operation. If it collapses to $|1\rangle$, then apply the unitary operation $\sigma^x$ to flip the state from $|1\rangle$ to $|0\rangle$. We see that applying one of two different unitaries ($I$ or $\sigma^x$) conditioned on the outcome of a measurement gives us a new capability beyond that of unitary operations.

### 4.3.2 Using CNOT gates to measure two-qubit operators

Given the ability to readout (i.e., measure) the states of individual qubits, how do we parlay this ability into the ability to make joint measurements of multi-qubit operators like $Z1Z_2$. As we have emphasized, measuring $Z_1Z_2$ produces only a single classical bit of information and causes different state collapse than measuring the individual values of $Z_1$ and $Z_2$ and then multiplying the results of the two measurements. We need a way to learn the value of $Z_1Z_2$ without learning the value of $Z_1$ or $Z_2$. Recall that to understand the probabilities of different measurement outcomes, we need to know the eigenvalues and eigenvectors of the operator being measured. Since both $Z_1$ and $Z_2$ are diagonal in the standard basis, so is their product. Thus any standard basis state is an eigenvector

$$
\begin{aligned}
Z_1Z_2|00\rangle &= (+1)|00\rangle && (4.26)\\
Z_1Z_2|01\rangle &= (-1)|01\rangle && (4.27)\\
Z_1Z_2|10\rangle &= (-1)|10\rangle && (4.28)\\
Z_1Z_2|11\rangle &= (+1)|11\rangle. && (4.29)
\end{aligned}
$$

Because the measurement result is always an eigenvalue of the operator being measured, we see that measurement of $Z_1Z_2$ tells us whether the two qubits are the same state or opposite states, but tells us nothing about which state the individual qubits are in.

Consider the circuit shown in Fig. 4.3. It uses two CNOT gates (here represented equivalently as controlled X (CX) gates). If $q_1$ and $q_2$ are the same, $q_0$ is flipped either 0 times (if $|b_1b_2\rangle = |00\rangle$) or two times (if $|b_1b_2\rangle = |11\rangle$). Either way, $q_0$ ends up in $b_0 = 0$. If $q_1$ and $q_2$ are the opposite, exactly one of the CX gates is activated and $q_0$ ends up in $b_0 = 1$. Thus measurement of $Z_0$ is equivalent to measurement of the joint operator $Z_1Z_2$. This scheme is readily extended to produce measurements of arbitrary strings of Pauli operators of any length. A related method will be discussed in Sec. 666666666 for executing multi-qubit gates (as opposed to measurements).

### 4.3.3 Using CNOT gates to produce entanglement

Now that we have established the CNOT unitary and how it acts on the standard basis, let us consider what happens when the control qubit is in

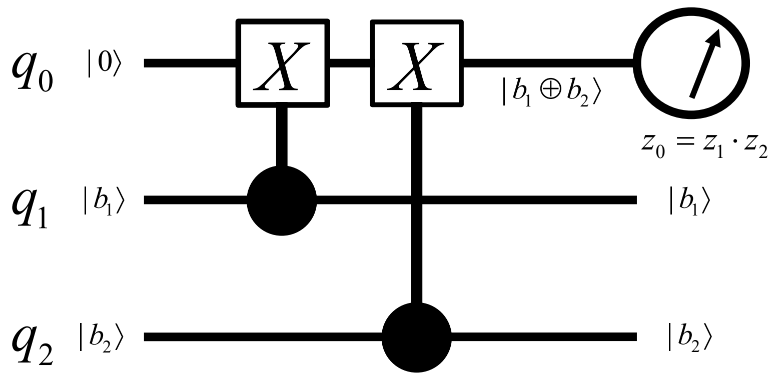

Figure 4.3: Circuit for mapping the joint operator $Z_1Z_2$ onto the auxiliary qubit operator $Z_0$. Measurement of $Z_0$ yields the value of $Z_1Z_2$ without conveying any information about the individual values of $Z_1$ and $Z_2$. $b_0, b_1, b_2 \in \{0, 1\}$ denote qubit states in the standard basis and $\oplus$ denotes bitwise addition mod 2.

a superposition state. We begin with a so-called *product state* or *separable state* which can be written as the product of a state for the first qubit and a state for the second qubit

$$\begin{aligned}|\Psi\rangle &= [\alpha_1|0\rangle + \beta_1|1\rangle] \otimes [\alpha_0|0\rangle + \beta_0|1\rangle] \\ &= \alpha_1\alpha_0|00\rangle + \alpha_1\beta_0|01\rangle + \beta_1\alpha_0|10\rangle + \beta_1\beta_0|11\rangle. \qquad (4.30)\end{aligned}$$

For example consider the following separable initial state

$$|\psi\rangle = |-\rangle|0\rangle = |\leftarrow\rangle|0\rangle = \frac{1}{\sqrt{2}}\left(|0\rangle - |1\rangle\right)|0\rangle \qquad (4.31)$$

$$= \frac{1}{\sqrt{2}}\left(|00\rangle - |10\rangle\right). \qquad (4.32)$$

When we apply the CNOT gate to this superposition state, the target qubit both flips and doesn't flip (depending on the state of the control)! The two qubits become 'entangled'

$$|B_2\rangle = \text{CNOT}_1|\psi\rangle = \frac{1}{\sqrt{2}}\left(|00\rangle - |11\rangle\right), \qquad (4.33)$$

where we have added a subscript to CNOT to explicitly indicate that qubit 1 is the control qubit. An *entangled state* of two qubits[2] is any state which

[2]The concept of entanglement is more difficult to uniquely define and quantify for $N > 2$ qubits.

cannot be written in separable form as a tensor product of single particle states.

**Box 4.3. LOCC No-Go Theorem** In order to convert a product state into an entangled state, we must use an entangling gate. Such gates are always controlled gates such as the CNOT or CPHASE. These gates cannot be written as a single Kronecker product of two single-qubit operators. There is an important no-go theorem which states that using local operations and classical communication (LOCC) two parties cannot establish an entangled state between them starting from product states. Here local operations means operations of the form $Z \otimes I$ or $I \otimes H$, which one party carries out independently of the other. Operations like CNOT and CPHASE, being conditional operations are explicitly not local. Classical communication refers to the two parties communicating with each other via a classical (i.e., without exchanging qubits) about which gates (or measurements) they have performed or request that the other part perform.

In order to perform a non-local gate, the qubits have to be (at some point in the protocol) physically nearby so they can interact with each other, or if remote, then the two parties much communicate via a quantum channel. For example, Alice can locally use a CNOT gate on a pair of qubits to create an entangled pair and then send one of the qubits to Bob.

A convenient basis in which to represent entangled states is the so-called

Bell basis[3]

$$|B_0\rangle = \frac{1}{\sqrt{2}}\left[|01\rangle - |10\rangle\right] \tag{4.34}$$

$$|B_1\rangle = \frac{1}{\sqrt{2}}\left[|01\rangle + |10\rangle\right] \tag{4.35}$$

$$|B_2\rangle = \frac{1}{\sqrt{2}}\left[|00\rangle - |11\rangle\right] \tag{4.36}$$

$$|B_3\rangle = \frac{1}{\sqrt{2}}\left[|00\rangle + |11\rangle\right]. \tag{4.37}$$

Each of these four states is a 'maximally entangled' state, but they are mutually orthogonal and therefore must span the full four-dimensional Hilbert space. Thus linear superpositions of them can represent any state, including even product states. For example,

$$|0\rangle|0\rangle = |00\rangle = \frac{1}{\sqrt{2}}\left[|B_2\rangle + |B_3\rangle\right]. \tag{4.38}$$

Entanglement is very mysterious and entangled states have many peculiar and counter-intuitive properties. In an entangled state the individual spin components have zero expectation value and yet the spins are strongly correlated. For example in the Bell state $|B_0\rangle$,

$$\langle B_0|\vec{\sigma}_1|B_0\rangle = \vec{0} \tag{4.39}$$

$$\langle B_0|\vec{\sigma}_0|B_0\rangle = \vec{0} \tag{4.40}$$

$$\langle B_0|\sigma_1^x\sigma_0^x|B_0\rangle = -1 \tag{4.41}$$

$$\langle B_0|\sigma_1^y\sigma_0^y|B_0\rangle = -1 \tag{4.42}$$

$$\langle B_0|\sigma_1^z\sigma_0^z|B_0\rangle = -1. \tag{4.43}$$

This means that the spins have quantum correlations which are stronger than is possible classically. In particular,

$$\langle B_0|\vec{\sigma}_1\cdot\vec{\sigma}_0|B_0\rangle = -3 \tag{4.44}$$

[3]Named after John Bell, the physicist who in the 1960's developed deep insights into the issues surrounding the concept of entanglement that so bothered Einstein. Bell proposed a rigorous experimental test of the idea that the randomness in quantum experiments is due to our inability to access hidden classical variables. At the time this was a theorist's 'gedanken' experiment, but today the 'Bell test' has rigorously ruled out the possibility of hidden variables. Indeed, the Bell test is now a routine engineering test to make sure that your quantum computer really is a quantum computer, not a classical computer.

despite the fact that in any product state $|\psi\rangle = |\hat{n}_1\rangle \otimes |\hat{n}_0\rangle$

$$|\langle\vec{\sigma}_1\rangle| = |\hat{n}_1| = |\langle\vec{\sigma}_0\rangle| = |\hat{n}_0| = 1. \tag{4.45}$$

**Box 4.4. Joint Measurements of Multiple Qubits** One might ask how the computer architect designs the hardware to make joint measurements of operators of the form $M = \sigma_1^x \sigma_0^z$, etc. First, a general remark: If she wants to measure $M = AB$, the product of two observables $A$ and $B$, then $M$ must be an observable (i.e., Hermitian).

$$\begin{aligned} M^\dagger &= (AB)^\dagger = B^\dagger A^\dagger = BA \\ M^\dagger &= M \iff AB = BA \iff [A, B] = 0, \end{aligned}$$

where $[A, B] \equiv AB - BA$ is the commutator of the two operators. Thus the product of two observables is not itself an observable unless the two operators commute. If $A$ and $B$ act on different qubits, then they automatically commute and they share the same eigenvectors (though not necessarily the same eigenvalues of course). This makes the Born rule easy to apply

$$|\psi\rangle = \sum_j \psi_j |j\rangle,$$

where $|j\rangle$ is the $j$th eigenvector of $A$,$B$, and $M = AB$ having eigenvalues

$$\begin{aligned} A|j\rangle &= a_j |j\rangle \\ B|j\rangle &= b_j |j\rangle \\ M|j\rangle &= a_j b_j |j\rangle. \end{aligned}$$

The computer architect might be able to devise a physical set up to directly make the joint measurement (see Fig. 4.3, but it is likely easier for her to employ her existing ability to make single-qubit measurements and thus measure $A$ and $B$ sequentially (or separately but in parallel). For simplicity, assume that the spectrum of each operator is non-degenerate (eigenvalues are unique). Measurement of $A$ yields the result $a_j$ with probability $|\psi_j|^2$ and the state collapses to $|j\rangle$. Subsequent measurement of $B$ yields the result $b_j$ with probability 1 since $|j\rangle$ is an eigenstate of $B$. Hence the measured value of $M$ formed by multiplying the two measurement results, $m_j = a_j b_j$ occurs with probability $|\psi_j|^2$, which is the same result that would have been obtained by measuring $B$ first and then $A$ or by directly measuring the joint operator $M = (AB)$. It is crucially important however to understand (as we discussed in Sec. 4.2 that joint measurements produce less information but induce less state collapse than measurements of the individual qubits involved in the joint measurement. This difference can be very useful.

We can obtain a useful picture of the Bell states by examining all of the possible two-spin correlators in the so-called 'Pauli bar plot' for the state $|B_0\rangle$ shown in Fig. 4.4. We see that all the single-spin operator (e.g. $IX$ and $ZI$) expectation values vanish. Because of the entanglement, each spin is, on average, totally unpolarized. Yet three of the two-spin correlators, $XX, YY, ZZ$, are all -1 indicating that the two spins are pointing in exactly opposite directions. This is becasue $|B_0\rangle$ is the rotationally invariant 'spin-singlet' state.

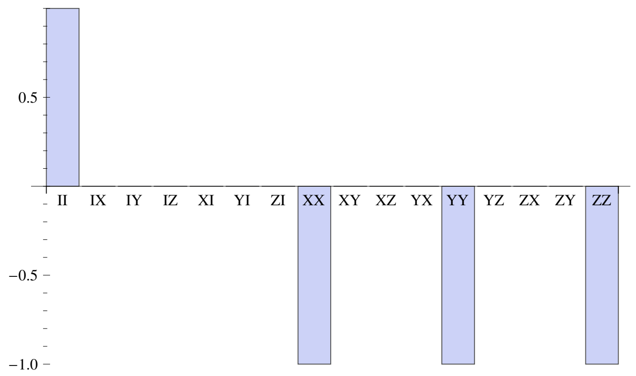

Figure 4.4: 'Pauli bar plot' of one and two spin correlators in the Bell state $|B_0\rangle$.

In maximally entangled states Bell states, the individual spins do not have their own 'state' in the sense that $\langle B|\vec{\sigma}_1|B\rangle = \vec{0}$, rather than $\hat{n}$, where $\hat{n}$ is some unit vector. We can only say for $|B_0\rangle$ for example that the $z$ components of the two spins will always be measured to be exactly opposite each other and have product $-1$. Remarkably, the same is true for the $x$ and $y$ components as well. Classical vectors of length 1 drawn from a random probability distribution simply cannot have this property. Thus the correlations of the spins in entangled states are intrinsically quantum in nature and stronger than is possible in any classical (or 'hidden variable' quantum) model.

**Exercise 4.3.** Derive Eqs. (4.39-4.45)

## 4.4 Bell Inequalities

We turn now to further consideration of the 'spooky' correlations in entangled states. We have already seen for the Bell state $|B_0\rangle$ that the components of the two spins are perfectly anti-correlated. Suppose now that Alice prepares two qubits in this Bell state and then sends one of the two qubits to Bob who is far away (say one light-year). Alice now chooses to make make a measurement of her qubit projection along some arbitrary axis $\hat{n}$. For simplicity let us say that she chooses the $\hat{z}$ axis. Let us further say that her measurement result is $-1$. Then she immediately knows that if Bob chooses to measure his qubit along the same axis, his measurement result will be the opposite, $+1$. It seems that Alice's measurement has collapsed the state of the spins from $|B_0\rangle$ to $|10\rangle$. This 'spooky action at a distance' in which Alice's measurement seems to instantaneously change Bob's distant qubit was deeply troubling to Einstein [3].

Upon reflection one can see that this effect cannot be used for superluminal communication (in violation of special relativity). Even if Alice and Bob had agreed in advance on what axis to use for measurements, Alice has no control over her measurement result and so cannot use it to signal Bob. It is true that Bob can immediately find out what Alice's measurement result was, but this does not give Alice the ability to send a message. In fact, suppose that Bob's clock were off and he accidentally made his measurement slightly before Alice. Would either he or Alice be able to tell? The answer is no, because each would see a random result just as expected. This must be so because in special relativity, the very concept of simultaneity is frame-dependent and not universal.

Things get more interesting when Alice and Bob choose different measurement axes. Einstein felt that quantum mechanics must somehow not be a complete description of reality and that there might be 'hidden variables' which if they could be measured would remove the probabilistic nature of the quantum description. However in 1964 John S. Bell proved a remarkable inequality [4] showing that when Alice and Bob use certain different measurement axes, the correlations between their measurement results are stronger than any possible classical correlation that could be induced by (local) hidden variables. Experimental violation of the Bell inequality proves that it is *not* true that quantum observables have values (determined by hidden classical variables) before we measure them. Precision experimental measurements which violate the Bell inequalities are now the strongest proof that quantum mechanics is correct and that local hidden variable theories are excluded.

Perhaps the simplest way to understand this result is to consider the CHSH inequality developed by Clauser, Horn, Shimoni and Holt [5] following Bell's ideas. Consider the measurement axes shown in Fig. 4.5. The experiment consists of many trials of the following protocol. Alice and Bob share a pair of qubits in an entangled state. Alice randomly chooses to measure the first qubit using $X$ or $Z$ while Bob randomly chooses to measure the second qubit using $X'$ or $Z'$ which are rotated 45 degrees relative to Alice's axes. After many trials (each starting with a fresh copy of the entangled state), Alice and Bob compare notes (via classical subluminal communication) on their measurement results and compute the following correlation function

$$S = \langle XX' \rangle + \langle ZZ' \rangle - \langle XZ' \rangle + \langle ZX' \rangle. \tag{4.46}$$

By correlation function of two measurement results we mean (for example)

$$\langle XZ' \rangle = \langle B_0 | XZ' | B_0 \rangle. \tag{4.47}$$

Experimentally this is measured by the following. For the $XZ'$ correlator, Alice and Bob select those instances in which Alice happens to have chosen to measure $X$ and Bob happens to have chosen to measure $Z'$. (This occurs say $N$ times, corresponding to 25% of the total number of trials on average.) Let $x_j = \pm 1$ and $z'_j = \pm 1$ be respectively Alice's random measurement result for $X$ in the $j$th trial and and Bob's random measurement result for $Z'$ in the $j$th trial. Then Alice and Bob's comparision of measurement results can be used to create the following unbiased (but noisy) *estimator* of the correlation

$$\langle B_0 | XZ' | B_0 \rangle \approx \frac{1}{N} \sum_{j=1}^{N} x_j z_j. \tag{4.48}$$

In the limit of large $N$ the statistical uncertainty in the estimator goes to zero and we obtain an arbitrarily accurate estimate of the correlation. The four possible combinations of the two measurement results are illustrated in Fig. 4.6. The correlator is given by the probabilities of the four outcomes

$$\langle B_0 | XZ' | B_0 \rangle = P_{++} - P_{-+} + P_{--} - P_{+-}. \tag{4.49}$$

If the measurements are perfectly correlated ($x_j = z'_j$ every time) then the correlator will be $+1$. If perfectly anticorrelated ($x_j = -z'_j$ every time), then the correlator will be $-1$. If the measurements are uncorrelated, then all four measurement outcomes will be equally likely and $x_j z'_j$ will be fully random ($\pm 1$ with equal probability) and the correlator will vanish (on average).

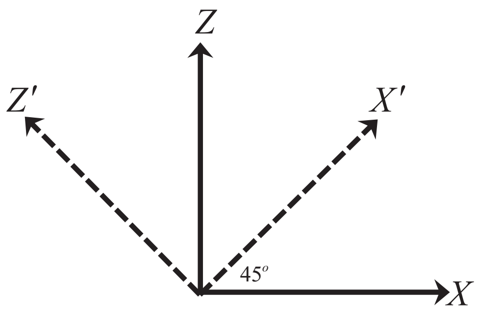

Figure 4.5: Measurement axes used by Alice (solid lines) and Bob (dashed lines) in establishing the Clauser-Horn-Shimoni-Holt (CHSH) inequality.

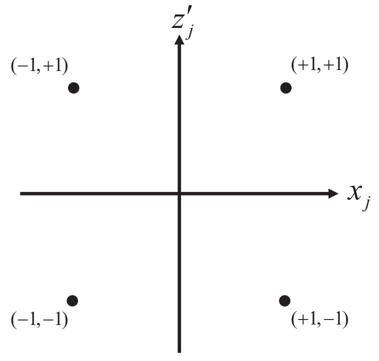

Figure 4.6: Illustration of the four possible measurement outcomes in the $j$th run of an experiment in which Alice measures $X$ and Bob measures $Z'$. The net correlator of the two measurements is given by Eq. (4.49).

Eq. (4.46) can be rewritten

$$S = \langle (X + Z)X' \rangle - \langle (X - Z)Z' \rangle. \tag{4.50}$$

Alice and Bob note that their measurement results are random variables which are always equal to either $+1$ or $-1$. In a particular trial Alice chooses randomly to measure either $X$ or $Z$. If you believe in the hidden variable theory, then surely, the quantities not measured still have a value of either $+1$ or $-1$ (because when we do measure them, they always are either $+1$ or $-1$). If this is true, then either $X = Z$ or $X = -Z$. Thus either $X + Z$ vanishes or $X - Z$ vanishes in any given realization of the random variables. The combination that does not vanish is either $+2$ or $-2$. Hence it follows immediately that $S$ is bounded by the CHSH inequality

$$-2 \leq S \leq +2. \tag{4.51}$$

It turns out however that the quantum correlations between the two spins in the entangled pair violate this classical bound. They are stronger than can ever be possible in any classical local hidden variable theory. To see this, note that because $\vec{\sigma}$ is a vector, we can resolve its form in one basis in terms of the other via

$$\sigma^{x'} = \frac{1}{\sqrt{2}}[\sigma^z + \sigma^x] \tag{4.52}$$

$$\sigma^{z'} = \frac{1}{\sqrt{2}}[\sigma^z - \sigma^x]. \tag{4.53}$$

Thus we can express $S$ in terms of the 'Pauli bar' correlations

$$S = \frac{1}{\sqrt{2}}\left[\langle XX + XZ + ZX + ZZ\rangle - \langle XZ - XX - ZZ + ZX\rangle\right]. \tag{4.54}$$

For Bell state $|B_0\rangle$, these correlations are shown in Fig. 4.4 and yield

$$S = -2\sqrt{2}, \tag{4.55}$$

in clear violation of the CHSH inequality. Strong violations of the CHSH inequality are routine in modern experiments.[4] This teaches us that in our quantum world, observables do not have values if you do not measure them. There are no hidden variables which determine the random values. The unobserved spin components simply do not have values. Recall that $\sigma^x$ and $\sigma^z$ are incompatible observables and when we choose to measure one, the other remains not merely unknown, but unknowable.

It is ironic that Einstein's arguments that quantum mechanics must be incomplete because of the spooky properties of entanglement have led to the strongest experimental tests verifying the quantum theory and falsifying all

[4]Although strictly speaking, in many experiments there are loopholes associated with imperfections in the detectors and the fact that Alice and Bob typically do not have a space-like separation. However recent experiments have closed all these loopholes. Note that all measured correlators have statistical uncertainty in them when the number of measurements $N$ is finite. However modern experiments can achieve results that exceed the Bell bound by many standard deviations.

local hidden variable theories. We are forced to give up the idea that physical observables have values before they are observed.

**Exercise 4.4.** Other Bell inequalities.

1. Work out the 'Pauli bar plots' (analogous to Fig. 4.4) for each of the Bell states $B_1, B_2, B_3$.
2. Using the same quantization axes as in Fig. 4.5, find the analog of Eq. (4.46) for the correlators that should be measured to achieve violation of the Bell inequality for these other Bell states.

## 4.5 Quantum Dense Coding

Now that we have learned about Bell states describing the entanglement of a pair of qubits, we will show that entanglement is a powerful resource for quantum information tasks. In this section we will see how quantum entanglement can be used as a resource to help in the task of quantum communication.

We saw above that quantum correlations are strong enough to violate certain classical bounds, but the spooky action at a distance seemed unable to help us send signals. We will learn more about this 'no signaling' condition in this section. It turns out that by using a special 'quantum dense coding' protocol, we can use entanglement to help us transmit information in a way that is impossible classically [2]. As preparation for understanding this protocol, let us consider the following situation. Alice has qubit $q_0$ and Bob has qubit $q_1$ and the combined system is in a product state. As an example, suppose that this state is

$$|\psi\rangle = |\chi\rangle \otimes |0\rangle. \tag{4.56}$$

If Alice and Bob are far apart, Alice is unable to do any operations on Bob's qubit, but she can perform a local unitary $U_0 = I \otimes U$ on her qubit $q_0$. We can now ask the following question: Given this constraint of local operations, how many distinct (i.e., orthogonal) states for the combined system can Alice reach starting from this initial state? Clearly for $U_0 = I \otimes X$, she can produce

$$(I \otimes X)|\psi\rangle = |\chi\rangle \otimes |1\rangle, \tag{4.57}$$

which is orthogonal to the original state. This however is the only orthogonal state Alice can reach. The two-qubit Hilbert space is four-dimensional but Alice cannot fully explore it. It seems 'obvious' that this is because she has no access to Bob's qubit, however as we will see, things become much less obvious when we consider entangled states.

The situation is very different for two-qubit entangled states. We take as our basis the four orthogonal Bell states in Eqs. (4.34-4.37). Suppose that Alice prepares the Bell state $|B_0\rangle$ using qubits $q_1, q_0$ and sends $q_1$ to Bob who is in a distant location. Using a remarkable protocol called quantum dense coding [2], Alice can now send Bob two classical bits of information by performing a local gate on $q_0$ and then sending it to Bob. The protocol (whose circuit in illustrated in Fig. 4.7) relies on the amazing fact that Alice can transform the initial Bell state into any of the other Bell states by purely local operations on her remaining qubit without communicating with Bob. The four possible unitary operations Alice should perform are $I_0, X_0, Y_0, Z_0$ which yield[5]

$$\begin{aligned} I_0|B_0\rangle &= +|B_0\rangle && (4.58)\\ Z_0|B_0\rangle &= -|B_1\rangle && (4.59)\\ X_0|B_0\rangle &= +|B_2\rangle && (4.60)\\ Y_0|B_0\rangle &= -i|B_3\rangle. && (4.61) \end{aligned}$$

It seems somehow miraculous that without touching Bob's qubit, Alice can reach all four orthogonal states by merely rotating her own qubit. However the fact that this is possible follows immediately from Eqs. (4.39) and (4.40), the Pauli bar plot in Fig. 4.4 and the corresponding plots for the other Bell states. In every Bell state, the expectation value of every component of $\vec{\sigma}_0$ (and $\vec{\sigma}_1$) vanishes. Thus for example

$$\langle B_0|\sigma_0^x|B_0\rangle = \langle B_0|I \otimes X|B_0\rangle = 0. \tag{4.62}$$

But this can only occur if the state $(\sigma_0^x|B_0\rangle)$ is orthogonal to $|B_0\rangle$! This in turn means that there are four possible two-bit messages Alice can send by associating each with one of the four operations[6] $I_0, X_0, Y_0, Z_0$ as shown in

[5]As usual we are simplifying the notation. For example $Z_0$ stands for the more mathematically formal $I \otimes Z$ since it applies $Z$ to Alice's qubit, $q_0$, and the identity to Bob's qubit, $q_1$. Note that the global phase factors are irrelevant to the workings of the protocol.

[6]The particular operators associated with each of the four binary numbers is somewhat arbitrary and was chosen in this case to correspond to a particular choice of Bob's decoding circuit which will be described later.

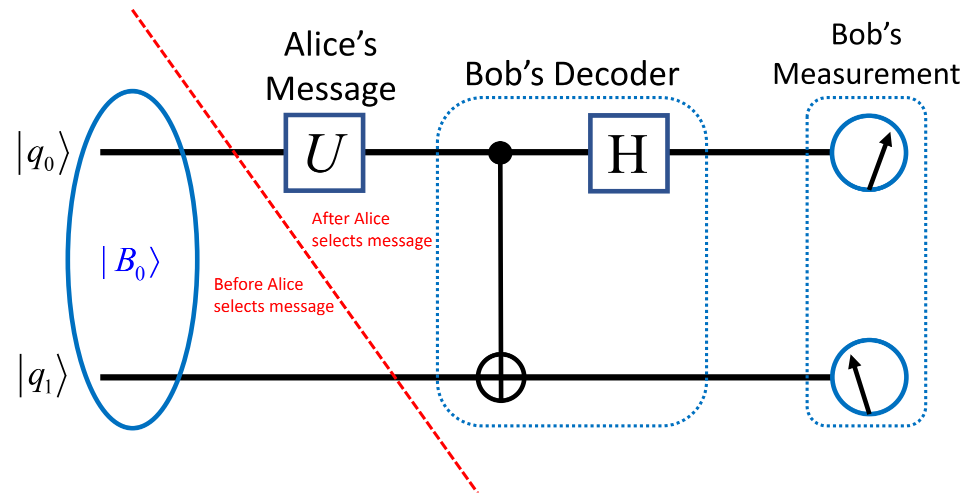

Figure 4.7: Illustration of the quantum dense coding protocol showing the power of having a prepositioned Bell pair shared by Alice and Bob. Alice has two qubits, $q_1, q_0$ and on Monday prepares them in an entangled Bell state $|B_0\rangle$. She sends $q_1$ to Bob. On Tuesday she decides to send Bob a two-bit classical message by choosing one of four possible unitaries $U \in \{I, X, Y, Z\}$ to apply to her remaining qubit, $q_0$. This unitary maps the initial Bell state $|B_0\rangle$ to one of the four orthogonal Bell states. She then sends $q_0$ to Bob who decodes the Bell pair (by mapping the four Bell basis states back to the standard computational basis). If Alice chooses the classical message $z_1, z_0$, Bob's decoder outputs a state $e^i\phi(z_1, z_0)|z_1z_0\rangle$, where $\phi(z_1, z_0)$ is an irrelevant phase factor. Bob then measures the qubits and obtains from the measurements two classical numbers $z_1, z_0$ corresponding to Alice's message.

Table 4.2.

The reader might now reasonably ask the following question. Before Alice sends Bob $q_0$ can he, by making local measurements on his qubit $q_1$, detect the fact that Alice's operation has changed the joint state of their two qubits? Clearly from the Holevo bound (see Box 2.1) he can learn at most 1 bit of information so would not be able to fully learn which of the four operations Alice did, but perhaps he could learn something. If he could, then special relativity would be violated because of signal would have passed instantaneously from Alice to Bob exceeding the bound set by the speed of light. The answer is a firm no, as discussed in Box 4.5.

| Alice's message | Alice's operation | State Bob receives | Bob's decoded state | Bob's measurement result $(z_1, z_0)$ |
|---|---|---|---|---|
| 00 | $Y_0$ | $-i\|B_3\rangle$ | $-i\|00\rangle$ | $(0,0)$ |
| 01 | $X_0$ | $+\ \|B_2\rangle$ | $+\ \|01\rangle$ | $(0,1)$ |
| 10 | $Z_0$ | $-\ \|B_1\rangle$ | $-\ \|10\rangle$ | $(1,0)$ |
| 11 | $I_0$ | $+\ \|B_0\rangle$ | $-\ \|11\rangle$ | $(1,1)$ |

Table 4.2: Illustration of the quantum dense encoding protocol, showing the unitary operation Alice must carry out on her qubit, $q_0$, the state produced by Bob's decoder and the measurement results Bob obtains from which he reads Alice's two classical bit message. The extra phase factors in front of the Bell states have no effect on the measurement results because Bob never has to deal with a superposition of these Bell states.

**Box 4.5. No Signaling Condition** We saw in Box 4.3 that it is impossible to create entanglement using LOCC. With the quantum dense coding protocol we see that if entanglement already exsits between two parties, it is possible for (either) one of the parties to change the entangled state using only LOCC. The 'no signaling' condition tells us that if one party changes the entangled state, the other party is not able to detect this using only local operations and measurements. Thus it is impossible for one party to communicate with the other using only local operations. To see why, note from Eqs. (4.39) and (4.40) for Bell state $|B_0\rangle$, and the analogous equations for the other Bell states, that every Pauli operator that one measures has zero expectation value, meaning that one sees only random shot noise and no signal. It is true that if both parties measure the same Pauli operator there will be perfect correlations in the results. However the results are purely random and in order to see the correlations, the two parties must 'compare notes' which requires classical communication at (or below) the speed of light. Equivalently, since Alice's measurement results are random and not under her control, she cannot use these as intentional messages to Bob. [Note for Pauli measurements which can only yield $\pm 1$, the mean of the distribution of measurement results, $m = p_+ - p_-$, determines the entire distribution, since $p_+ + p_- = 1$. Thus if the mean is unchanged, the entire distribution is unchanged. Hence no messages can be hidden in the distribution if the mean cannot be changed.]

The upshot of all this is that, while it may appear that there is spooky action at a distance in the changes that one party can make in an entangled state using LOCC, these changes are locally invisible to the other party because only the correlations in measurement results of the two parties actually change. Computation of these correlations requires (subluminal) classical communication.

**Exercise 4.5.** The argument for the no-signaling condition given in Box 4.5 relies on the quantum state being one of the 4 Bell states that Alice can create using the quantum dense coding protocol by applying a Pauli gate to her qubit. We know however that superpositions of Bell states can be used to represent ordinary product states. Show that if, instead of applying a Pauli gate like $Y_0$, Alice applies a rotation by angle $\theta$

$$U = e^{-i\frac{\theta}{2}Y_0} = \cos\theta\,[I \otimes I] - i\sin\theta\,[I \otimes Y],$$

the resulting superposition of Bell states still obeys the no-signaling condition for Bob's measurements.

After encoding her message by carrying out the appropriate operation on her qubit, Alice physically transmits her qubit to Bob. Bob then makes a joint measurement (details provided further below) on the two qubits which tells him which of the four Bell states he has and thus recovers two classical bits of information even though Alice sent him only one quantum bit after deciding what the message was. The pre-positioning of the entangled pair has given them a resource which doubles the number of classical bits that can be transmitted with one (subsequent) quantum bit. Of course in total Alice transmitted two qubits to Bob, so the Holevo bound has not been violated. The key point is that the first qubit was sent in advance of deciding what the message was. How weird is that!?

This remarkable protocol sheds considerable light on the concerns that Einstein raised in the EPR paradox [3]. It shows that the special correlations in Bell states can be used to communicate information in a novel and efficient way by 'prepositioning' entangled pairs shared by Alice and Bob. However causality and the laws of special relativity are not violated because Alice still has to physically transmit her qubit(s) to Bob in order to send the information.

The above protocol requires Bob to determine which of the four Bell states he has. How can he do this? A theorist might suggest that Bob simply measure the operator

$$M_{\text{Bell}} = 0|B_0\rangle\langle B_0| + 1|B_1\rangle\langle B_1| + 2|B_2\rangle\langle B_2| + 3|B_3\rangle\langle B_3|. \tag{4.63}$$

The four eigenstates of this Hermitian operator are the four (orthonormal) Bell states, and we see explicitly that Bell state $j$ has eigenvalue $j$. Thus the measurement result tells us precisely which Bell state the system is in.

**Exercise 4.6.** Prove that measurement of the operator $M_{\text{Bell}}$ can be used to non-deterministically create entangled states. If you measure $M_{\text{Bell}}$ in the product state $|00\rangle$, what entangled states will result and with what probabilities will they occur? Repeat your calculation for the product state $|01\rangle$. Note that having randomly created one of the four Bell states, you can convert it to any other via single qubit Pauli gates on either one of the two qubits as illustrated in Eqs. (4.58-4.61).

An experimentalist would find it more practical to use the quantum circuit shown in Fig. 4.8 that uses two standard gates to uniquely map each of the Bell states onto one of the four computational basis states (eigenstates of $Z_1$ and $Z_2$) that are easily measured. The first symbol indicates the CNOT gate which flips the target qubit (in this case qubit 1) if and only if the control qubit (qubit 0) is in the excited state. The second gate in the circuit (denoted H) is the Hadamard gate that was introduced in Sec. 3.4.1 acting on qubit zero.

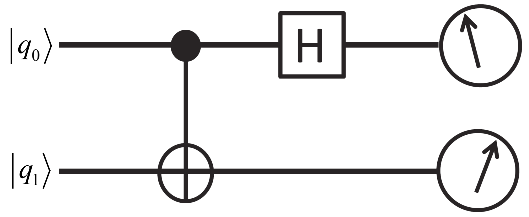

Figure 4.8: Bell-basis measurement circuit comprising a Bell state decoder with a CNOT and Hadamard gate followed by measurement in the computational basis. This circuit permits measurement of which Bell state a pair of qubits is in by mapping the states to the standard basis of eigenstates of $\sigma_0^z$ and $\sigma_1^z$.

**Exercise 4.7.** Prove the following identities for the circuit shown in Fig. 4.8 where qubit 0 is the control and qubit 1 is the target

$$\mathrm{H}_0\,\mathrm{CNOT}_0|B_0\rangle = -|11\rangle \tag{4.64}$$
$$\mathrm{H}_0\,\mathrm{CNOT}_0|B_1\rangle = +|10\rangle \tag{4.65}$$
$$\mathrm{H}_0\,\mathrm{CNOT}_0|B_2\rangle = +|01\rangle \tag{4.66}$$
$$\mathrm{H}_0\,\mathrm{CNOT}_0|B_3\rangle = +|00\rangle \tag{4.67}$$

Once Bob has mapped the Bell states onto unique computational basis states, he measures $Z_0$ and $Z_1$ separately, thereby gaining two bits of classical information and effectively reading the message Alice has sent as shown in the last column of Table 4.2. Note that the overall sign in front of the basis states produced by the circuit is irrelevant and not observable in the measurement process. Also note that to create Bell states in the first place, Alice can simply run the circuit in Fig. 4.8 backwards.

**Exercise 4.8.** Construct explicit quantum circuits that take the starting state $|00\rangle$ to each of the four Bell states.)

## 4.6 No-Cloning Theorem Revisited

Now that we have understood the EPR paradox and communication via quantum dense coding, we can gain some new insight into the no-cloning theorem (see Box 3.4 and [1]). It turns out that if cloning of an unknown quantum state were possible, then we could use an entangled pair of qubits to communicate information at superluminal speeds in violation of special relativity. Consider the following protocol. Alice and Bob share an entangled Bell pair in state

$$|B_0\rangle = \frac{1}{\sqrt{2}}[|01\rangle - |10\rangle]. \tag{4.68}$$

Alice chooses to measure her qubit (say $q_0$) in either the $Z_0$ basis or the $X_0$ basis. The choice she makes defines one classical bit of information. The result of the measurement collapses the entangled pair into a simple product state. If Alice chooses the $Z_0$ basis for her measurement, then the state collapses with equal probability to either $|01\rangle$ (if the measurement result is $-1$) or $|10\rangle$ (if the measurement result is $+1$). In the resulting product state, Bob's qubit will have the opposite value for $Z_1$.

Suppose instead that Alice chooses to measure in the $X_0$ basis. To analyze this situation, note that the $B_0$ state can also be written in terms of the X eigenstates as

$$|B_0\rangle = \frac{1}{\sqrt{2}}[|-+\rangle - |+-\rangle]. \tag{4.69}$$

Then in the product state produced by the measurement, Bob's qubit will be either $|+\rangle$ or $|-\rangle$.

How does Bob tell whether his state is an $X$ eigenstate or a $Z$ eigenstate? With only one qubit, he cannot tell for sure since a measurement of $X$ or

$Z$ always yields an $X$ or $Z$ eigenstate. Suppose for example the state Bob has happens to be $|0\rangle$ (because Alice measured $Z$ and got $-1$). Then if Bob measures $X$ he learns nothing at all (his result is completely random). If Bob measures $Z$ his result will be $+1$. He learns from this that his state could have been $|0\rangle$, or $|+\rangle$, or $|-\rangle$. However it could *not* have been $|1\rangle$ since that would have yielded a measurement of $-1$. Thus Bob gains a small amount of information. See Box 4.6 for how to quantify this information gain.

If Bob can clone his qubit to make many copies then he can distinguish the four cases with near unit probability. If he measures a large number of copies in the $Z$ basis and always gets the same answer, he knows that his qubit is almost certainly in a $Z$ eigenstate. If even a single measurement result is different from the first, he knows his qubit cannot be in a $Z$ eigenstate and so must be in an $X$ eigenstate. (Of course he could also measure a bunch of copies in the $X$ basis and gain the same information.)

Let us suppose as a specific example, that Alice makes her measurement in the $Z_0$ basis. Further suppose that Bob clones his qubit to obtain $2N$ copies. He measures $N$ of the qubits in the $Z_1$ basis and $N$ in the $X_1$ basis. Every single one of his measurements in the $Z_1$ basis will give precisely the same result (the opposite of whatever Alice got in her one measurement) because (after Alice's measurement) the resulting cloned state is an eigenstate of $Z_1$. Because the state is an eigenstate of $Z_1$, measurements of $X_1$ will be completely random with equal probability of $\pm 1$ results. The probability that all $N$ of the $X_1$ measurements would be all be $+1$ or all be $-1$ is $P_\pm(N) = 2^{-N}$. Thus the probability that Bob would be unable to decide which basis gave identical measurements and which gave random would be

$$P_{\mathrm{fail}} = 2 \cdot 2^{-N}. \tag{4.70}$$

Since the failure probability falls exponentially with $N$, the procedure is efficient. In case of a 'tie' between the two measurement strings, Bob could simply clone a few more copies (from one qubit he has set aside and not measured) and with high probability would break the tie.

Note that Alice cannot control her measurement result, only the choice of measurement axis. Thus Bob does not gain any information from the sign of his measurement result (which is opposite that of Alice if he measures in the same basis as Alice). He learns one bit of information about the quantization axis of Alice's measurement (if he can clone his qubit) from the measurement results all being the same, but it does not matter what the sign of these measurement results is. Since measurement state collapse is instantaneous, this would yield superluminal communication that would work across any spatial distance and would not take any time to occur (beyond the time it takes Bob to clone his qubit a few times) and thus would violate special relativity. Hence cloning is fundamentally incompatible with relativistic causality.

In fact, cloning would make it possible for Alice to transmit an unlimited number of classical bits using only a single Bell pair. Alice could choose an arbitrary measurement axis $\hat{n}$. The specification of $\hat{n}$ requires two real numbers (the polar and azimuthal angles). It would take a very large number of bits to represent these real numbers to some high accuracy. Now if Bob can make an enormous number of copies of his qubit, he can divide the copies in three groups and measure the vector spin polarization $\langle\vec{\sigma}\rangle$ to arbitrary accuracy. From this he knows the polarizaton axis $\hat{n} = \pm\langle\vec{\sigma}\rangle$ Alice chose (up to an unknown sign since he does not know the sign of Alice's measurement result for $\hat{n} \cdot \sigma$). Hence Bob has learned a large number of classical bits of information. The accuracy $\epsilon$ (and hence the number of bits $\sim \log_2(1/\epsilon)$) is limited only by the statistical uncertainties resulting from the fact that his individual measurement results can be random, but these can be reduced to an arbitrarily low level with a sufficiently large number of copies $N \sim 1/\epsilon^2$ of the state. Note however that the 'cost' $N$ is exponential in the number of bits accurately learned. It would be remarkably useful to transmit multiple classical bits using a single quantum bit. But, it turns out to be impossible because of the no-cloning theorem.

**Exercise 4.9.** Suppose that Bob has the ability to clone his qubit, but does not have the ability to make perfect measurements. Let there be a probability $0 < \epsilon < 0.5$ that his measuring apparatus produces a result that is the opposite of the true result each time he makes a measurement (i.e., a measurement of $Z$ in state $|0\rangle$ sometimes yields $-1$ instead of the correct $+1$). Suppose he uses $N$ copies of his qubit to measure $Z$ and $N$ to measure $X$ in order to determine the quantization axis Alice chose. The failure probability will naturally be higher than that given in Eq. (4.70). For $N \gg 1$, give an estimate of Bob's failure probability for determining the quantization axis of the qubit.

**Box 4.6. Quantifying Information Gain from a measurement** Suppose that by prior agreement, Alice randomly gives Bob either the state $|0\rangle$ or the state $|1\rangle$ with equal probability. The Shannon information entropy in this probability distribution is

$$S_{\text{prior}} = -\sum_{j=0,1} p_j \log_2 p_j = 1\,\text{bit}$$

for the case $p_0 = p_1 = \frac{1}{2}$. Bob knows he should measure in the $Z$ basis. After Bob makes his measurement, we need to update the probability distribution based on the new information received. (See the discussion of Bayes Rule in App. A.) For example, if Bob's measurement is $z = +1$ then the new probability distribution is $p_0 = 1, p_1 = 0$ since there is no randomness left. The Shannon entropy $S_{\text{post}} = 0$. Hence the information gained by Bob from his measurement is given by the decrease in randomness of the distribution upon measurement

$$I = S_{\text{prior}} - S_{\text{post}} = 1\,\text{bit},$$

as expected.

The situation is more complex if by prior agreement Alice randomly gives Bob one copy of $|0\rangle, |1\rangle, |+\rangle$, or $|-\rangle$ with equal probability. Here of course $S_{\text{prior}} = 2\,\text{bits}$. Suppose Bob measures in the $Z$ basis and obtains the result $z = +1$. Application of Bayes Rule (see App. A) yields the new posterior distribution of states consistent with the measurement result

$$p_{|0\rangle} = \frac{1}{2}, \quad p_{|1\rangle} = 0, \quad p_{|+\rangle} = \frac{1}{4}, \quad p_{|-\rangle} = \frac{1}{4},$$

which yields

$$S_{\text{post}} = -\frac{1}{2}\log_2\frac{1}{2} - \frac{1}{4}\log_2\frac{1}{4} - \frac{1}{4}\log_2\frac{1}{4} = \frac{3}{2}\,\text{bits}.$$

Thus the information gain Bob receives from his measurement is

$$I = S_{\text{prior}} - S_{\text{post}} = \frac{1}{2}\,\text{bit}.$$

## 4.7 Quantum Teleportation

Even though it is impossible to clone an unknown quantum state, it is possible for Alice to 'teleport' an unknown state to Bob as long as her copy of the original is destroyed in the process [6]. Just as for quantum dense coding, teleportation protocols take advantage of the power of 'pre-positioned' entangled pairs. However unlike quantum dense coding where Alice ultimately sends her qubit to Bob, teleportation only requires Alice to send two classical bits to Bob. She does not have to send any additional quantum bits to Bob after pre-positioning the initial Bell pair. A simple protocol is as follows (see Fig. 4.9): Alice creates a $|B_3\rangle$ Bell state and sends one of the qubits to Bob. Alice has in her possession an additional qubit in an unknown state

$$|\psi\rangle = \alpha|0\rangle + \beta|1\rangle \tag{4.71}$$

which she wishes Bob to be able to obtain without her physically sending it to him. Again, this must done at the expense of destroying the state of her copy of the qubit because of the no-cloning theorem.

Remarkably, Alice is able to teleport her (unknown) state to Bob using only LOCC, provided that she and Bob share a pre-existing Bell pair. Alice applies the Bell state measurement protocol illustrated in Fig. 4.8 to determine the joint state of the unknown qubit and her half of the Bell pair she shares with Bob. She then transmits two *classical bits* using an ordinary classical communication channel relaying her measurement results to Bob (via the double line wires shown in Fig. 4.8 ). Note that even if Alice knew what the state $|\psi\rangle$ was, two classical bits alone are not enough to provide Bob with the information needed to prepare his own copy of the state $|\psi\rangle$ (since it takes an infinite number of bits to specify the two angles determining the position on the Bloch sphere).

To see how Bob is able to reconstruct the initial state using the pre-existing Bell pair, note that we can rewrite the initial state of the three qubits in the basis of Bell states for the two qubits that Alice will be measuring as

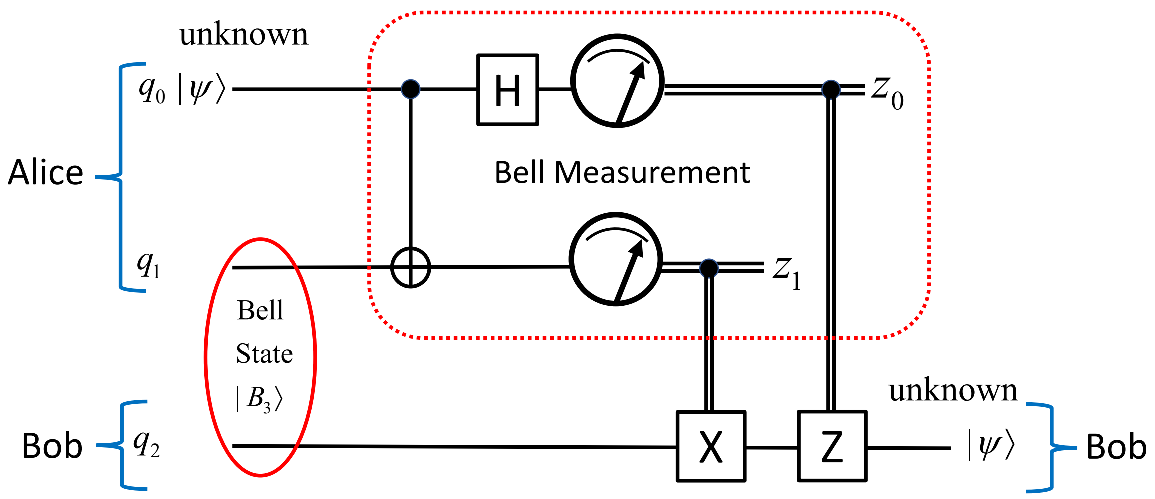

Figure 4.9: Quantum circuit that Alice can use to teleport and unknown state to Bob using only two bits of classical information, provided that she and Bob have shared a Bell pair in advance (in this case Bell state $|B_3\rangle$). Single wires indicate quantum channels (qubits), double wires indicate classical information channels. Based on the measurement results $z_0$ and $z_1$ that Alice sends to Bob, Bob performs the operation $Z^{z_0}X^{z_1}$ on his half of the Bell pair ($q_2$). That is, if Alice's measurement result $z_1 = 1$ he performs an $X$ gate and then if Alice's measurement result $z_0 = 1$, he performs a $Z$ gate. The quantum state of Alice's qubit is destroyed (randomly collapsed) by the measurements and so the no-cloning theorem is obeyed.

follows

$$|\Phi\rangle = \underbrace{|B_3\rangle}_{q_2q_1} \otimes \underbrace{[\alpha|0\rangle + \beta|1\rangle]}_{q_0} \tag{4.72}$$

$$= \frac{\alpha}{\sqrt{2}}[|000\rangle + |110\rangle] + \frac{\beta}{\sqrt{2}}[|001\rangle + |111\rangle] \tag{4.73}$$

$$\begin{aligned} &= \frac{1}{2}\underbrace{[\beta|0\rangle - \alpha|1\rangle]}_{q_2} \otimes \underbrace{|B_0\rangle}_{q_1q_0} + \frac{1}{2}\underbrace{[\beta|0\rangle + \alpha|1\rangle]}_{q_2} \otimes \underbrace{|B_1\rangle}_{q_1q_0} \\ &+ \frac{1}{2}\underbrace{[\alpha|0\rangle - \beta|1\rangle]}_{q_2} \otimes \underbrace{|B_2\rangle}_{q_1q_0} + \frac{1}{2}\underbrace{[\alpha|0\rangle + \beta|1\rangle]}_{q_2} \otimes \underbrace{|B_3\rangle}_{q_1q_0} \end{aligned} \tag{4.74}$$

From this representation we see that when Alice tells Bob which Bell state she found (from the decoding step), Bob can find a local unitary operation to perform on his qubit to recover the original unknown state. The appropriate operations are

| Alice's Bell state ($q_1q_0$) | Alice's decoded state ($q_1q_0$) | Bob's operation ($q_2$) |
|---|---|---|
| $\|B_0\rangle$ | $-\|11\rangle$ | $ZX$ |
| $\|B_1\rangle$ | $+\|10\rangle$ | $X$ |
| $\|B_2\rangle$ | $+\|01\rangle$ | $Z$ |
| $\|B_3\rangle$ | $+\|00\rangle$ | $I$ |

Notice that the final (decoded) state of Alice's two qubits contains no information about the quantum amplitudes $\alpha, \beta$ in Alice's original state. This information has been destroyed during the teleportation process and hence the no-cloning theorem is not violated when Bob obtains a 'copy' of Alice's state, since he has the only 'copy.'

Notice also the similarities between quantum dense coding and state teleportation. Both use a pre-positioned Bell pair. In quantum dense coding Alice uses one of four operations, $I, X, Y, Z$ on her qubit to send two classical bits to Bob (which become available when she sends her qubit to Bob and he measures the pair in the Bell basis to obtain one of four results). In teleportation, Alice makes a joint measurement of her unknown state $|\psi\rangle$ and her half of the Bell pair to obtain one of four results and sends the resulting two classical bits to Bob who then uses that information to apply one of four unitaries to his half of the Bell pair to reconstruct Alice's state.

How might quantum teleportation be useful in a quantum computer architecture? It is much easier to transmit classical bits than quantum bits from one part of the computer to another (or between nodes in a quantum computer cluster). Suppose you have a slow and unreliable channel for transmitting quantum bits that can be used to (slowly and unreliably) distribute Bell pairs. There exist error-correction protocols to distill a few high-fidelity Bell pairs from a larger collection of faulty Bell pairs. See the discussion to be added in Chapter XXXXX. Using this we can preposition high-quality Bell pairs between distant nodes and then teleport quantum states between them using only 'local operations and classical communication' (LOCC).

We will later study an even more powerful protocol in which quantum gates on distant qubits can be performed locally and then teleported into a distant qubit. Like state teleportation, this 'gate teleportation' has numerous applications in quantum computer architectures. For example, the CNOT logic gate requires an operation on the target qubit conditioned on the state of the control qubit. This means the two qubits have to physically interact in some way, something that is most easily accomplished if the qubits are adjacent to each other in the hardware layout. We can release this constraint and peform a CNOT between distant qubits using gate teleportation.

## 4.8 YET TO DO:

1. Explain how Hadamard gates interchange control and target in the CNOT.

2. Show that there is NO state for which Alice can change the probability of the outcomes for Bob's Z measurement by applying any unitary to her qubit. (Because operators on different qubits commute.) However Alice can change Bob's probability by making a measurement of her qubit. But, she can't control the outcome of the measurement, so there is no superluminal communication.

3. The value measured for the observable is always one of the eigenvalues of that observable and the state always collapses to the corresponding eigenvector. (If two or more of the eigenvalues are degenerate, then the situation is slightly more subtle. The state is projected onto the degenerate subspace and then normalized.) Thus if the measurement result is the non-degenerate eigenvalue $\lambda_j$ then the state $|\psi\rangle$ collapses to
$$|\psi\rangle \to \frac{P_j|\psi\rangle}{\sqrt{\langle\psi|P_j|\psi\rangle}},$$
where the square root in the denominator simply normalizes the collapsed state. Introduce this projector idea for measurements to avoid the confusion that some students think measurement of $X$ is represented by multiplying by $X$. Discuss further the confusion that Pauli operators are both unitary operations and Hermitian observables.

   If one measures $\vec{\sigma}\cdot\hat{n}$ (i.e. asks 'Are you in state $|+\hat{n}\rangle$ or state $|-\hat{n}\rangle$?), then from the Born rule, the probability that the measurement result is $\pm 1$ is $|\langle\pm\hat{n}|\psi\rangle|^2$. If you make a measurement of a product state, say $|00\rangle$ that asks, 'Which entangled Bell state are you in?' the state will collapse to one of the Bell states and the collapse (aka 'measurement back action') leaves the state entangled. Note this doesn't work with the Bell state measurement that relies on the decoding into the standard basis.

# Chapter 5

# Algorithms and Complexity Classes

The very first quantum algorithm was proposed by Deutsch in 1985 and so it will be the first that we study. This was followed by the Deutsch-Jozsa algorithm in 1992, the first algorithm that was exponentially faster than any deterministic classical algorithm. We will learn however that probabilistic classical algorithms can be much faster than deterministic ones, provided that one can except a small chance of failure. Motivated by this, Simon invented a problem for which even probabilistic classical algorithms take exponentially long time and he found a fast quantum algorithm with bounded failure probability that runs in polynomial time.

These 'toy' problems and quantum algorithms were expressly invented to be difficult for classical computers and easy for quantum computers in order to demonstrate the possibilities of quantum hardware. They are not practically useful algorithms but they paved the way for all subsequent ones which have been invented, including most famously, the Grover search algorithm for unstructured databases and Shor's algorithm for finding the prime factors of large numbers–a task that is required to break RSA public key encryption. We will study a key component of Shor's algorithm, the quantum Fourier transform, since it has wide application.

Before beginning our study of algorithms we will need to understand two key concepts: 'phase kickback' of a controlled unitary operation and the concept of a quantum oracle.

## 5.1 Phase 'Kickback' of a Controlled-Unitary Operation

Recall that the NOT operation ($X$) looks very classical in the standard basis

$$\begin{aligned} \text{NOT}|0\rangle &= |1\rangle & (5.1) \\ \text{NOT}|1\rangle &= |0\rangle. & (5.2) \end{aligned}$$

NOT simply reproduces the classical truth table. The same can be said for the controlled-NOT (CNOT) operation which applies NOT to a target qubit iff the control qubit is in $|1\rangle$.

When we put the control bit of a CNOT circuit (see Fig. 5.1) into a superposition state, we obtain a new non-trivial quantum effect, entanglement

$$\text{CNOT}_0|0\rangle|+\rangle = \frac{1}{\sqrt{2}}[|00\rangle + |11\rangle]. \tag{5.3}$$

(Recall that the subscript on the CNOT tells us which qubit is the control.)

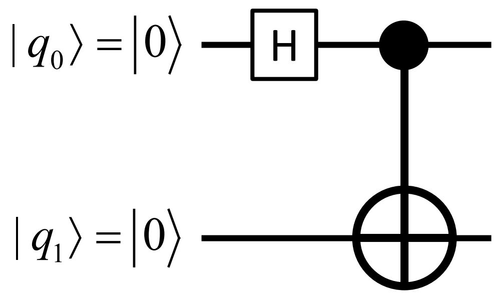

Figure 5.1: Circuit applying a CNOT gate with the control qubit $q_0$ being in a superposition state so that the initial product state is mapped onto an entangled Bell state.

Something else interesting happens when the quantum NOT gate is applied when both control and target are in superposition states as illustrated in Fig. 5.2. Notice that $|\pm\rangle$ are eigenstates of the NOT operation

$$\begin{aligned} X|+\rangle &= (+1)|+\rangle & (5.4) \\ X|-\rangle &= (-1)|-\rangle. & (5.5) \end{aligned}$$

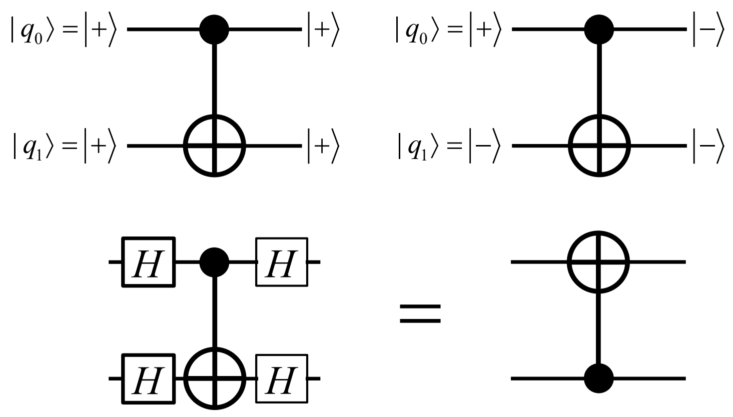
$|q_0\rangle = |+\rangle$
$|+\rangle$
$|q_1\rangle = |+\rangle$
$|+\rangle$
$|q_0\rangle = |+\rangle$
$|-\rangle$
$|q_1\rangle = |-\rangle$
$|-\rangle$
H
H
H
H
=

Figure 5.2: Upper panels: Circuit applying a CNOT gate with the control qubit $q_0$ and the target qubit $q_1$ being in a superposition state. Because the target is an eigenstate of the NOT operation, it is actually the control qubit that gets flipped (due to the phase kickback)! Lower panels: Using the Hadamard gates to change from the $Z$ basis to the $X$ basis interchanges the control and target of the CNOT gate.

Here we see a very non-classical effect: we can flip a qubit and leave it in exactly the same state (up to a possible global phase factor)! It turns out that if we do a controlled NOT operation on $|\pm\rangle$ the eigenvalue of NOT is a *relative* phase (not a global phase) that gets 'kicked back' onto the control. The circuits shown in Fig. 5.2 illustrate the peculiar effect that results. It seems that in the $X$ basis the role of target and control are reversed!

$$\big(H \otimes H\big)\mathrm{CNOT}_0\big(H \otimes H\big) = \mathrm{CNOT}_1. \tag{5.6}$$

That this reversal is due to the phase kickback can be seen from the following

$$\begin{aligned}
\mathrm{CNOT}_0|++\rangle &= \frac{1}{2}[|0\rangle + |1\rangle] \otimes |0\rangle + \frac{1}{2}(X \otimes I)\big[[|0\rangle + |1\rangle] \otimes |1\rangle\big] \\
&= |++\rangle \qquad (5.7)\\
\mathrm{CNOT}_0|-+\rangle &= \frac{1}{2}[|0\rangle - |1\rangle] \otimes |0\rangle + \frac{1}{2}(X \otimes I)\big[[|0\rangle - |1\rangle] \otimes |1\rangle\big] \\
&= \frac{1}{2}[|0\rangle - |1\rangle] \otimes |0\rangle - \frac{1}{2}\big[-[|0\rangle + |1\rangle] \otimes |1\rangle\big] \\
&= |--\rangle \qquad (5.8)
\end{aligned}$$

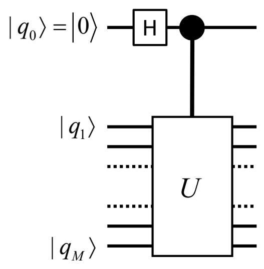

Figure 5.3: Circuit applying a general controlled unitary to the state of $M$ qubits. If the $M$ qubits are in a eigenstate of $U$ with eigenvalue $e^{i\varphi}$, the phase kickback on the control rotates it around the $z$ axis by angle $\varphi$.

The CNOT gate is a particular example of the general concept of a controlled unitary operation, illustrated in Fig. 5.3. If the control qubit ($q_0$) is 0, then the controlled unitary applies the identity. If the control qubit is 1, then the $M$-qubit unitary $U$ is applied to qubits 1 through $M$. In this circuit the control qubit $q_0$ is placed in the $|+\rangle$ state by the Hadamard gate.

We are used to thinking about the (real-valued) eigenvalues and eigenvectors of Hermitian observables. Here it is useful to think about the eigenvalues and eigenvectors of the unitary operator $U$. Because $U^\dagger U = I$, it must be that the eigenvalues of any unitary lie on the unit circle in the complex plane. Thus in the basis of eigenvectors $|\psi_j\rangle$ of $U$ with corresponding eigenvalues $e^{i\varphi_j}$, the matrix representation of $U$ has the form

$$U = \begin{pmatrix} e^{i\varphi_1} & 0 & 0 & \cdots \\ 0 & e^{i\varphi_2} & 0 & \cdots \\ 0 & 0 & e^{i\varphi_3} & \cdots \\ \vdots & \vdots & \vdots & \ddots \end{pmatrix}. \tag{5.9}$$

We see immediately that if qubits 1 through $M$ are in the eigenstate $|\psi_j\rangle$, the phase kicked back onto the control qubit is $\varphi_j$

$$\mathrm{C}U[\underbrace{|\psi_j\rangle}_{\text{target}} \otimes \underbrace{(\alpha|0\rangle + \beta|1\rangle)}_{\text{control}}] = |\psi_j\rangle \otimes (\alpha|0\rangle + \beta e^{i\varphi_j}|1\rangle). \tag{5.10}$$

This phase kickback causes (and thus can be detected by) a rotation of the control qubit by an angle $\varphi_j$ around the $z$ axis. For the case of the controlled-NOT gate studied above, unitary is $X$ and its eigenvalues are $\pm 1$, so the phase kickback is either 0 or $\pi$.

Phase kickback will play a central role in a number of quantum algorithms that we will study including the quantum Fourier transform and the phase estimation algorithm.

## 5.2 Exponentiation Gadget

The exponentiation 'gadget' is the circuit illustrated in the left panel of Fig. 5.4 exponentiates a unitary matrix $U$ to carry out the unitary gate

$$e^{-i\frac{\theta}{2}U} = \cos\frac{\theta}{2}I - i\sin\frac{\theta}{2}U, \tag{5.11}$$

provided that $U$ is also Hermitian so that $U = U^\dagger$ and thus $U^2 = I$. Multi-qubit products of Pauli operators are natural examples of unitaries satisfying this condition. Interestingly, despite being rotated in the middle of the circuit, the control qubit starts in 0 and ends in 0, merely playing the role of a 'catalyst.'

Single qubit rotations around an axis on the Bloch sphere defined by an arbitrary unit vector $\hat{n}$

$$e^{-i\frac{\theta}{2}\hat{n}\cdot\vec{\sigma}} \tag{5.12}$$

satisfy the Euler-Pauli identity in Eq. (5.11) and are generally natively available on quantum computing hardware. Hence we don't usually need a special exponentiation gadget to execute such gates. (Note that the exponentiation gadget in fact uses one $X_\theta$ gate which is a single qubit rotation a round the $x$ axis.) However if $U$ is a more complex (Hermitian) unitary such as the three-qubit operation $U = Z \otimes X \otimes Z$, the ability to exponentiate it is unlikely to be natively available on the hardware. If however we are able to execute a controlled version of this unitary (conditioned on the state of a separate control qubit), then we can take advantage of the exponentiation gadget in Fig. 5.4.

To see in more detail how the exponentiation gadget works, let $|\psi\rangle$ represent the state of qubits 1 through $M$. The state of the system after the first controlled unitary is

$$\frac{1}{\sqrt{2}}\Big[|\psi\rangle \otimes |0\rangle + \big(U|\psi\rangle\big) \otimes |1\rangle\Big]. \tag{5.13}$$

Using the Euler-Pauli identity, we see that after the $X_\theta$ gate the state of the system is

$$\frac{1}{\sqrt{2}}\Big[|\psi\rangle\otimes[\cos\frac{\theta}{2}|0\rangle - i\sin\frac{\theta}{2}|1\rangle] + \big(U|\psi\rangle\big)\otimes[\cos\frac{\theta}{2}|1\rangle - i\sin\frac{\theta}{2}|0\rangle]\Big]. \tag{5.14}$$

Using the fact that $U^2 = I$, we see that after the second controlled unitary, the state of the system is

$$\frac{1}{\sqrt{2}}\Big[|\psi\rangle\otimes[\cos\frac{\theta}{2}|+\rangle + \big(U|\psi\rangle\big)\otimes[-i\sin\frac{\theta}{2}|+\rangle]\Big] = \Big[\big[\cos\frac{\theta}{2}I - i\sin\frac{\theta}{2}U\big]|\psi\rangle\Big]\otimes|0\rangle, \tag{5.15}$$

which reproduces Eq. (5.11). Notice that for $\theta = 0$, this is the identity gate. This makes sense because in this case, the $X_\theta$ gate in Fig. 5.4 is the identity and the two controlled unitaries either act zero times or twice, in both cases yielding identity for the overall circuit.

As a simple but practical example, suppose that $U$ is the three-qubit unitary $U = Z\otimes Z\otimes Z$. By itself, this operator cannot take a product state into an entangled state. For example

$$U\left(|+\rangle\otimes|+\rangle\otimes|+\rangle\right) = |-\rangle\otimes|-\rangle\otimes|-\rangle \tag{5.16}$$

takes a product state to a product state. The fact that we end up with a product state remains true even if we exponentiate the individual gates since

$$\begin{aligned}&\left(e^{-i\frac{\theta_2}{2}Z}\otimes e^{-i\frac{\theta_1}{2}Z}\otimes e^{-i\frac{\theta_0}{2}Z}\right)[|+\rangle\otimes|+\rangle\otimes|+\rangle] = \\ &\left(e^{-i\frac{\theta_2}{2}Z}|+\rangle\right)\otimes\left(e^{-i\frac{\theta_1}{2}Z}|+\rangle\right)\otimes\left(e^{-i\frac{\theta_0}{2}Z}|+\rangle\right).\end{aligned} \tag{5.17}$$

However, this product of exponentials is not the same as the exponential of the product, since by the Pauli-Euler identity

$$ZZZ(\theta) = e^{-i\frac{\theta}{2}(Z\otimes Z\otimes Z)} = \cos\frac{\theta}{2}\,I - i\sin\frac{\theta}{2}\,(Z\otimes Z\otimes Z) \tag{5.18}$$

which (for generic $\theta$) is an entangling gate for qubits $q_1, q_2, q_3$ in the circuit shown in Fig. 5.4

$$\begin{aligned}&ZZZ(\theta)\,[|+\rangle\otimes|+\rangle\otimes|+\rangle] = \\ &\cos\frac{\theta}{2}\,[|+\rangle\otimes|+\rangle\otimes|+\rangle] - i\sin\frac{\theta}{2}\,[|-\rangle\otimes|-\rangle\otimes|-\rangle].\end{aligned} \tag{5.19}$$

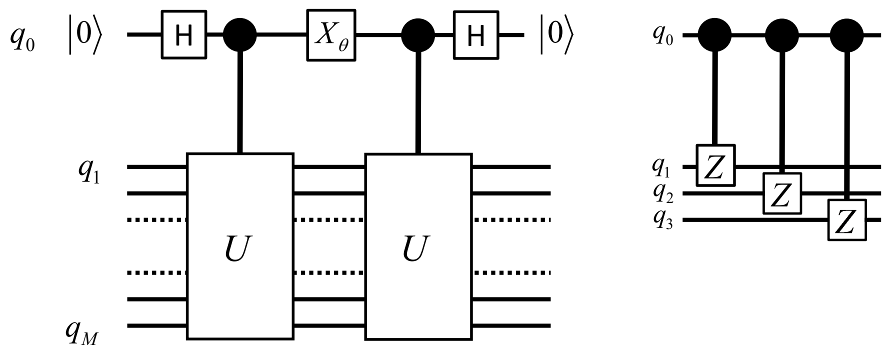

Figure 5.4: Left panel: Circuit representing the exponentiation gadget that exponentiates a unitary $U$ provided that it is also Hermitian, $U = U^\dagger$. The single-qubit gate $X_\theta \equiv e^{-i\frac{\theta}{2}X}$ is a rotation by angle $\theta$ about the $x$ axis. Right panel: circuit to synthesize the controlled ZZZ unitary $U$ using three cZ gates.

This superposition of two product states is an entangled state (for generic $\theta \neq 0, \pi$). The controlled version of this three-qubit unitary is readily achieved using three cZ gates as shown in the right panel of Fig. 5.4.

The exponentiation gadget can be used a variety of tasks, but is particularly useful for synthesizing complicated unitaries out of a series of simpler gates. Physicists familiar with the concept of time evolution under a Hamiltonian will recognize Eq. (5.18) as giving the time evolution operator for a spin model with three-spin interactions.

It is interesting to note that by inserting identity operations in the form of pairs of Hadamard gates (using the identities $H^2 = I$, $HXH = Z$, and $HZH = X$) and using the CNOT (equivalently CX) identity shown in the bottom panels of Fig. 5.2, we can give a new, and very different looking but fully equivalent, representation of the exponentiation gadget for the case $U = Z \otimes Z \otimes Z$ shown in Fig. 5.4. This representation, shown in Fig. 5.5 is intimately related to the circuit shown in Fig. 4.3 that allows for multi-qubit $Z$ measurements. The first portion of this gadget uses three CNOT (CX) gates to map the value of $Z_1Z_2Z_3$ onto $Z_0$ (just as was done in the measurement circuit in Fig. 4.3). The $Z_\theta$ gate then rotates $q_0$ by an angle $\theta$ about the $z$ axis. Then the last portion of the circuit 'uncomputes' the mapping returning $q_0$ to state $|0\rangle$. The $Z_\theta$ rotation in the middle of the circuit creates the desired exponential unitary given in Eq. (5.18). In general, any Hermitian unitary (e.g., an arbitrary Pauli operator string) automatically

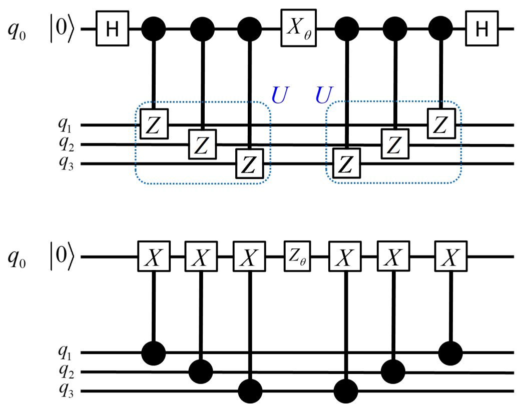

Figure 5.5: Upper panel: The exponentiation gadget shown in the left panel of Fig. 5.4, for the particular case of $U = Z_1Z_2Z_3$. Lower panel: A completely equivalent alternative circuit, reminiscent of the multiqubit measurement circuit in Fig. 4.3.

has only eigenvalues $\pm 1$ and can be mapped onto a single auxiliary qubit using this trick and thus any such unitary can be exponentiated.

## 5.3 Quantum Oracles

*Oracle, a person (such as a priestess of ancient Greece) through whom a deity is believed to speak.*
*From the Latin oraculum, from orare, 'to speak'*

A quantum oracle is a 'blackbox' whose inner workings *may* be inaccessible to the user. Oracles are often used as inputs to quantum algorithms. In particular they are used to bring classical data into a quantum computation and can also be used inside quantum algorithms to evaluate functions. We will soon learn about the Grover algorithm to search an unsorted classical database. The algorithm requires an oracle to make 'calls' to query the data base. Typically an oracle is a unitary operation (no measurements are involved that would bring in partial state collapse) as illustrated in Fig. 5.6.

As an example, let $\vec{b} = (b_{n-1}b_{n-2}\dots b_2b_1b_0)$ be a classical binary bit string of length $n$ and let $f$ be a classical binary function: $f(\vec{b}) = \vec{c}$, where $\vec{c}$ is also a binary bit string of the same length $n$. If we want, we can just think of this function as a bunch of classical data as illustrated in Table 5.1

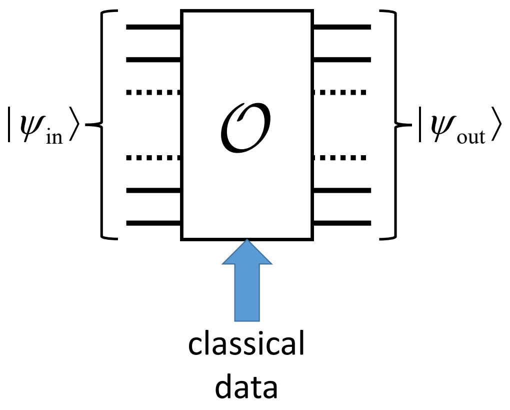

Figure 5.6: Circuit representation of a generic oracle $\mathcal{O}$ that performs a unitary transformation based on (i.e., parametrized by) classical data supplied to it..

| $\vec{b}$ | $\vec{c}$ |
|---|---|
| 000 | 001 |
| 001 | 010 |
| 010 | 011 |
| 011 | 100 |
| 100 | 101 |
| 101 | 110 |
| 110 | 111 |
| 111 | 000 |

Table 5.1: Data defining the binary function $f(\vec{b}) = \vec{c}$, that is equivalent to $F(k) = (k+1)$ mod 8 for integer $k \in \{0, 7\}$.

How do we turn this classical binary function into a quantum oracle? Let the quantum state $|\psi_{\text{in}}\rangle = |b_{n-1}b_{n-2}\dots b_2b_1b_0\rangle$ in the standard basis represent the input to the oracle (and to the function), and let the quantum state $|\psi_{\text{out}}\rangle = |c_{n-1}c_{n-2}\dots c_2c_1c_0\rangle$ represent the output of the oracle (and the function). Then the action that we require of the unitary oracle is

$$\mathcal{O}|\vec{b}\rangle = |\vec{c}\rangle = |f(\vec{b})\rangle. \tag{5.20}$$

This seems straightforward, but actually there is a problem if $f$ is not invertible (that is, $f(\vec{x}) = f(\vec{y})$ for some $\vec{x} \neq \vec{y}$). If this is the case then

$$\mathcal{O}|\vec{x}\,\rangle = \mathcal{O}|\vec{y}\,\rangle \tag{5.21}$$

implies that

$$\langle \vec{x}\,|\mathcal{O}^\dagger\mathcal{O}|\vec{y}\,\rangle = 1, \tag{5.22}$$

even though $\langle \vec{x}\,|\vec{y}\,\rangle = 0$ because $\vec{x} \neq \vec{y}$. Hence $\mathcal{O}$ cannot be unitary because it does not preserve inner products. Equivalently, it is not invertible and, by definition, unitaries are invertible since $U^{-1} = U^\dagger$.

Here is a second problem: the length $m$ of the output string for a generic function need not be the same as the length $n$ of the input string. We can fix this problem and at the same time guarantee that $\mathcal{O}$ is unitary by the circuit construction illustrated in Fig. 5.7. This circuit executes the transformation

$$\mathcal{O}(|\vec{d}\rangle \otimes |\vec{b}\rangle) = |\vec{d} \oplus f(\vec{b})\rangle \otimes |\vec{b}\rangle, \tag{5.23}$$

where $\oplus$ represents bitwise addition modulo 2 (bitwise XOR) as defined in Eq. (1.48). For example,

$$(11010111) \oplus (01011100) = (10001011). \tag{5.24}$$

This construction should look familiar because it is *exactly* analogous to that used in the middle panel of Fig. 1.6 to construct a reversible AND gate. The input to the function is available at the output so that the gate is reversible. Recalling the argument used for the reversible AND gate, notice that

$$\mathcal{O}^2(|\vec{d}\rangle \otimes |\vec{b}\rangle) = |\vec{d} \oplus f(\vec{b}) \oplus f(\vec{b})\rangle \otimes |\vec{b}\rangle = |\vec{d}\rangle \otimes |\vec{b}\rangle. \tag{5.25}$$

For the AND gate, the particular function used had two input bits and one output bit: $f(x, y) = x \wedge y$, but the reversibility argument applies to any binary function. This guarantees that $\mathcal{O}^2 = I$ and thus the inverse exists $\mathcal{O}^{-1} = \mathcal{O}$. This also shows that the only allowed eigenvalues of $\mathcal{O}$ are $\pm 1$.

Furthermore, if we represent $\mathcal{O}$ as a matrix acting on a column vector representing $|\vec{d}\rangle \otimes |\vec{b}\rangle$ it will yield a new vector representing $|\vec{d} \oplus f(\vec{b})\rangle \otimes |\vec{b}\rangle$. Similarly,

$$\mathcal{O}(|\vec{d} \oplus f(\vec{b})\rangle \otimes |\vec{b}\rangle) = |\vec{d}\rangle \otimes |\vec{b}\rangle, \tag{5.26}$$

shows us that the matrix representing $\mathcal{O}$ is symmetric. Because the quantum amplitudes of the states defined by the binary strings $\vec{d}, \vec{b}, f(\vec{b})$ are all taken to be real, it is straightforward to show that the matrix representing $\mathcal{O}$ is real and symmetric, and therefore it is Hermitian. Hence $\mathcal{O}^\dagger = \mathcal{O} = \mathcal{O}^{-1}$ proving that $\mathcal{O}$ is unitary. An alternative proof involves showing that $\mathcal{O}$ maps orthogonal states to orthogonal states. Let

$$|\psi\rangle = |\vec{d}\rangle \otimes |\vec{b}\rangle, \tag{5.27}$$

$$|\psi_f\rangle = |\vec{d} \oplus f(\vec{b})\rangle \otimes |\vec{b}\rangle, \tag{5.28}$$

$$|\psi'\rangle = |\vec{d}'\rangle \otimes |\vec{b}'\rangle, \tag{5.29}$$

$$|\psi'_f\rangle = |\vec{d}' \oplus f(\vec{b}')\rangle \otimes |\vec{b}'\rangle. \tag{5.30}$$

Then one can readily show that

$$\langle\psi'|\psi\rangle = 0 \implies \langle\psi'_f|\psi_f\rangle = 0. \tag{5.31}$$

For the case $\vec{b}' \neq \vec{b}$, the orthogonality follows immediately, independent of any property of the function $f$. For the case $\vec{b}' = \vec{b}$, $\vec{d}' \neq \vec{d}$, it follows from $f(\vec{b}) = f(\vec{b}')$ that $|\psi_f\rangle$ and $|\psi'_f\rangle$ involve different bit strings $\vec{d} \oplus f(\vec{b})$ and $\vec{d}' \oplus f(\vec{b})$ and are therefore orthogonal. This together with the eigenvalues of $\mathcal{O}$ being $\pm 1$ also proves unitarity.

Thus we have found a construction for creating a unitary oracle based on classical data supplied in the form of classical binary function $f$, even if that function is itself *not* reversible. The reversibility relies on $f(\vec{b}) \oplus f(\vec{b}) = \vec{0}$ which is true for any binary function.

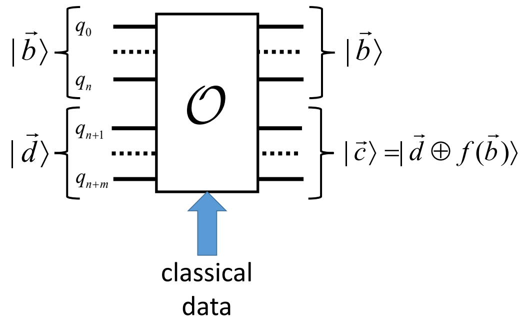

Figure 5.7: Circuit representation of an oracle representing a function whose domain (input) is bit strings of length $n$ and whose range (output) is bit strings of length $m$. $\oplus$ represents bitwise addition mod 2 (bitwise XOR). The oracle gate is guaranteed to be unitary and reversible even if the classical function $f$ is not.

Again, the simplest example of all this is the AND gate, or equivalently, the Toffoli gate (or CCNOT gate) shown in Fig. 1.6. This gate can be taken directly from the (reversible version of the) classical gate into the quantum gate construction shown in Fig.5.7 whose action is defined by Eq. (5.23).

**Exercise 5.1.** Construct a quantum oracle corresponding to the classical function that adds two one-bit binary numbers (a0) and (b0) to obtain one two-bit binary number (c1 c0).

## 5.4 Deutsch Algorithm

We are now ready to study our first algorithm. As mentioned in the introduction to this chapter, in 1985 David Deutsch developed the very first quantum algorithm. It only solves a 'toy' problem, but it opened the door to a new world. Let us now enter that new world.

Here is the problem setup: Alice creates an oracle for a binary function $f$ of her choosing that maps one bit to one bit (i.e., the input and output string lengths are $n = m = 1$). Bob's task is to query the oracle with his quantum computer to acquire information about the (unknown to him) function. Recall from the list in Table 1.3 and Eq. (1.41) that for $n = m = 1$, the number of such functions is $2^2 = 4$. Two of the functions (ONE and ZERO) have *constant* output and the other two (IDENTITY and NOT) have *balanced* outputs (i.e., 0 and 1 occur equally often).

Bob's task is not to find the full function encoded in the oracle (a task which would require learning two classical bits of information that distinguish the four different functions), but simply to learn one bit of information that tells him if the function is constant or balanced. Classically Bob must query the oracle twice to identify the full function $f$ before being able to determine which category the function is in. For example, suppose Bob asks the oracle, 'What is $f(0)$?' and the oracle replies '1.' As a result Bob knows that either $f =$ ONE (constant) or $f =$ NOT (balanced), but he does not know which. Bob is therefore forced to make a second query, 'What is $f(1)$,' to resolve the uncertainty. The answer tells him exactly which function $f$ is and thus which category it is in. He has learned two bits of information which is more than he needed, but there was no way to avoid this.

If Bob has a quantum computer he can use the Deutsch algorithm to make a single query to the oracle but with a 'superposition of questions' that will, through quantum interference, give Bob the correct answer to the question of whether the function is constant or balanced. The circuit for this is given in the Fig. 5.8. To see how this works notice that Bob does not require the individual values of $f(0)$ and $f(1)$ but only needs to know one bit of global information about the function, namely the value of $g = f(0) \oplus f(1)$ since:

$$g = 0 \iff f \text{ is CONSTANT} \tag{5.32}$$

$$g = 1 \iff f \text{ is BALANCED.} \tag{5.33}$$

Again it is clear that classically one has to query the oracle twice to learn the value of this global property of the function. While this is only a toy problem, it is remarkable that one can learn a *global* property of a function with only a single (quantum) query!

Let us now see how the Deutsch quantum algorithm works. Since this is a quantum oracle, Bob is free to give a superposition state as input

$$|\psi_{\text{in}}\rangle = |d\rangle \otimes |+\rangle = |d\rangle \otimes \frac{1}{\sqrt{2}}\big(|0\rangle + |1\rangle\big), \tag{5.34}$$

where $d$ is the bit value at the input to the oracle (on what will be the eventual output line) as shown in Fig. 5.8. It follows from the linearity of unitary transformations that the output state is

$$|\psi_{\text{out}}\rangle = \frac{1}{\sqrt{2}}\big\{|d \oplus f(0)\rangle \otimes |0\rangle + |d \oplus f(1)\rangle \otimes |1\rangle\big\}. \tag{5.35}$$

We see that there is information about both $f(0)$ and $f(1)$ in the output state even though we have queried the oracle only once. We have asked a 'superposition of questions.'

Our task is to now harness this result to achieve Bob's goal. Unfortunately, if we stop here and measure the state in the computational basis, it will collapse randomly into *either* the state

$$|\psi_{\text{out}}\rangle = |d \oplus f(0)\rangle \otimes |0\rangle, \tag{5.36}$$

*or* the state

$$|\psi_{\text{out}}\rangle = |d \oplus f(1)\rangle \otimes |1\rangle. \tag{5.37}$$

From the measurement result (and knowing the initial state of the $d$ qubit) we will randomly learn either the value of $f(0)$ or the value of $f(1)$. Even though the state before the measurement contained information about both $f(0)$ and $f(1)$, the measurement has *not* captured the global information we seek.

To remedy this situation we need to put both the $b$ and $d$ input qubits into superposition

$$\begin{aligned} |\psi_{\rm in}\rangle &= |-\rangle \otimes |+\rangle \\ &= \frac{1}{2}\big\{|0\rangle\otimes|0\rangle + |0\rangle\otimes|1\rangle - |1\rangle\otimes|0\rangle - |1\rangle\otimes|1\rangle\big\}, \end{aligned} \tag{5.38}$$

for which the oracle yields

$$\begin{aligned} |\psi_{\rm out}\rangle = \frac{1}{2}\big\{&|0\oplus f(0)\rangle\otimes|0\rangle + |0\oplus f(1)\rangle\otimes|1\rangle \\ &-|1\oplus f(0)\rangle\otimes|0\rangle - |1\oplus f(1)\rangle\otimes|1\rangle\big\}. \end{aligned} \tag{5.39}$$

To make further progress, let us consider the two cases:

**Case I**: $f$ is constant.

In this case $f(0) = f(1)$ and $f(0)\oplus f(1) = 0$. Thus we can rewrite the output state as

$$|\psi_{\rm out}\rangle = \left(\frac{|0\oplus f(0)\rangle - |1\oplus f(0)\rangle}{\sqrt{2}}\right)\otimes\left(\frac{|0\rangle+|1\rangle}{\sqrt{2}}\right) \tag{5.40}$$

$$= (\pm|-\rangle)\otimes(|+\rangle)\,. \tag{5.41}$$

Note that because the function $f$ is constant, the state $|c\rangle$ is going to be $\pm|-\rangle$. The (unobservable) $\pm$ phase factor is determined by whether the constant function is ONE or ZERO. As we will see shortly, the information Bob seeks lies in the state $|b\rangle$.

**Case II**: $f$ is balanced.

In the case that $f$ is balanced, we know that $f(0)\oplus f(1) = 1$ which implies

$$0\oplus f(0) = 1\oplus f(1) \tag{5.42}$$

$$0\oplus f(1) = 1\oplus f(0). \tag{5.43}$$

Using these relations, Eq. (5.39) can be rewritten

$$|\psi_{\rm out}\rangle = \frac{1}{\sqrt{2}}\big[|0\oplus f(0)\rangle\otimes|-\rangle + |0\oplus f(1)\rangle\otimes(-|-\rangle)\big] \tag{5.44}$$

$$= (\pm|-\rangle)\otimes(|-\rangle)\,. \tag{5.45}$$

In this case, the (unobservable) $\pm$ global phase factor depends on whether the balanced function is IDENTITY or NOT. We see that, in contrast to Eq. (5.41), the state of $|b\rangle$ has now changed to the $|-\rangle$ state. This means that Bob will therefore be able to determine from a measurement of $b$ in the $x$ basis whether the function is balanced or constant, after only a single query of the oracle.

Before moving on, it will be useful to repeat the above analysis in a new way that will prove useful when we study the Deutsch-Jozsa algorithm. (The analysis here follows that in Nielsen and Chuang except they use the opposite ordering for the qubit labels.) The input state to the algorithmic circuit shown in Fig. 5.8 is $|-\rangle \otimes |+\rangle$ (in the number convention where the top wire is qubit $q_0$ and the bottom qubit is $q_1$). The state after the oracle acts is

$$\begin{aligned}|\psi_1\rangle &= \mathcal{O}_f \left\{ \frac{\big(|0\rangle - |1\rangle\big)}{\sqrt{2}} \otimes \frac{1}{\sqrt{2}} \sum_{x=0,1} |x\rangle \right\} \\ &= \sum_{x=0,1} \frac{1}{\sqrt{2}} \big(|0 \oplus f(x)\rangle - |1 \oplus f(x)\rangle\big) \otimes \frac{1}{\sqrt{2}} |x\rangle \\ &= \frac{1}{\sqrt{2}} \sum_{x=0,1} (-1)^{f(x)} |-\rangle \otimes |x\rangle \\ &= |-\rangle \otimes \frac{1}{\sqrt{2}} \sum_{x=0,1} (-1)^{f(x)} |x\rangle \end{aligned} \tag{5.46}$$

Note that the dummy variable $x$ here ranges over $\{0, 1\}$ and does not represent anything to do with eigenvectors of the Pauli $X$ operator.

To find the state after the final Hadamard gate, it is useful to invoke the following handy identity which can be readily verified by hand for the two cases $x = 0$ and $x = 1$

$$H|x\rangle = \frac{1}{\sqrt{2}} \sum_{z=0,1} (-1)^{xz} |z\rangle. \tag{5.47}$$

Using this identity, the output state of the circuit is

$$|\psi_2\rangle = |-\rangle \otimes \frac{1}{2} \sum_{z=0,1} \sum_{x=0,1} (-1)^{f(x)+xz} |z\rangle. \tag{5.48}$$

Let's look at the quantum amplitude $a_z$ to find qubit $q_0$ in state $|z\rangle$

$$a_0 = \frac{1}{2}\sum_{x=0,1}(-1)^{f(x)}, \tag{5.49}$$

$$a_1 = \frac{1}{2}\sum_{x=0,1}(-1)^{f(x)+xz}. \tag{5.50}$$

Notice that if $f$ is constant (so that $f(x) = f(0)$) we have

$$a_0 = (-1)^{f(0)} = \pm 1, \tag{5.51}$$

$$a_1 = (-1)^{f(0)}\frac{1}{2}\sum_{x=0,1}(-1)^x = 0. \tag{5.52}$$

For this case the measurement result is guaranteed to yield $z = 0$ since $|a_0|^2 = 1$. Conversely if $f$ is balanced, then the summation

$$\sum_{x=0,1}(-1)^{f(x)} = 0, \tag{5.53}$$

because the summand is $+1$ and $-1$ equally often. Thus $z = 0$ never occurs for the balanced case and the measurement result must always be $z = +1$. In fact, it is straightforward to show that for the balanced case the amplitude to end up in state $|-\rangle \otimes |1\rangle$ is

$$a_1 = \frac{1}{2}\sum_{x=0,1}(-1)^{f(x)+x} = (-1)^{f(0)} = \pm 1. \tag{5.54}$$

These results are perfectly consistent with Eqs. (5.41) and (5.45) once you take into account that the final Hadamard gate that is included in Fig. 5.8 but not present in Fig. 5.7.

**Exercise 5.2.** The oracle for the Deutsch algorithm encodes one of four possible functions, ZERO, ONE, IDENTITY, and NOT. Construct explicit quantum circuits to realize each of these oracle functions.

## 5.5 Deutsch-Jozsa Algorithm

The Deutsch algorithm showed us that it is possible to learn a global property of a function that maps one classical bit into another classical bit by

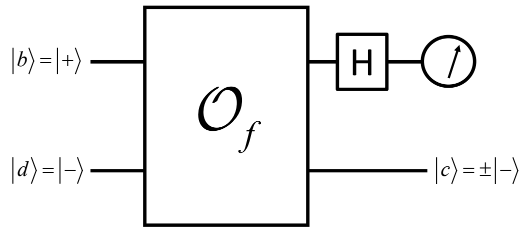

Figure 5.8: Circuit that executes the Deutsch algorithm to determine with a single query whether the function $f$ is constant or balanced. The classical data defining the function $f$ is encoded in the oracle $\mathcal{O}_f$. Measurement of qubit $b$ yielding result $z = +1$ tells us that $f$ is constant, while measurement result $z = -1$ tells us that $f$ is balanced.

encoding the function in a quantum oracle and querying the oracle only a single time. This is a toy model but a highly instructive one. The Deutsch-Jozsa algorithm also involves a toy problem, but a more sophisticated one, designed to show off an *exponential* separation between the power of the best deterministic classical algorithm for the problem and a simple quantum algorithm.

From our study of the Deutsch algorithm we know that there are four binary functions $f : \{0,1\} \to \{0,1\}$ and that two of them are constant and two of them are balanced. For Deutsch-Jozsa we will study functions that map $n$ bits to one bit: $f : \{0,1\}^n \to \{0,1\}$ where the situation is more complicated. Let $\vec{x}, \vec{y}$ be vectors in $\{0,1\}^n$ (i.e., binary strings of length $n$). Alice chooses a function $f$ such that $f(\vec{x}) = 0$ (say) for some $\vec{x}$ and $f(\vec{y}) = 1$ (say) for some $\vec{y}$. Recall that the number of functions on $n$ bits into $m$ bits is $Z(n,m) = 2^{m2^n}$. In this case $m = 1$ so $Z = 2^{2^n}$. Out of this enormous space of possible functions that Alice could choose, only two of them are constant,

$$\text{ZERO}: \ f(\vec{x}) = 0 \ \forall \vec{x}, \tag{5.55}$$

$$\text{ONE}: \ f(\vec{x}) = 1 \ \forall \vec{x}. \tag{5.56}$$

A balanced function is defined as before. Of the $2^n$ possible input strings, it outputs 0 for half of its $2^n$ possible arguments and outputs 1 for the other half of its arguments. If we look at the example of $n = 2$ which is listed in its entirety in Table 1.4, there are $Z(2,1) = 2^{2^2} = 16$ possible functions. Of these, two are constant, 6 are balanced and 8 are neither constant nor balanced. For larger $n$, the vast majority of the functions are neither constant nor balanced. (See Ex. 5.3).

In the Deutsch-Jozsa problem, Alice gives Bob a promise that the function $f$ she has chosen for her oracle maps $n$ bits to one bit and is either constant or balanced. (Note that since most functions are neither, this is an important promise.) Bob's task is to discover where the $f$ is constant or balanced The circuit for the Deutsch-Jozsa algorithm is essentially the same as in Fig. 5.8, except the upper wire is replaced by $n$ input wires in a product state in which each qubit in the $|+\rangle$ state. At the output, a Hadamard is applied to each of the upper wires followed by a $Z$ measurement of each of the $n$ upper wires.

We begin our analysis of the circuit by noting that the input product state of the upper $n$ wires an be written as an equal superposition of all computational basis states

$$|+\rangle^{\otimes n} = \frac{1}{\sqrt{2^n}} \sum_{\vec{x}} |\vec{x}\,\rangle, \tag{5.57}$$

where $\vec{x}$ is a binary string of length $n$ and the sum is over all $2^n$ such strings.

The analog of Eq. (5.46) for the state of the circuit after application of the oracle is thus

$$\begin{aligned} |\psi_1\rangle &= \mathcal{O}_f \left\{ \frac{\big(|0\rangle - |1\rangle\big)}{\sqrt{2}} \otimes \frac{1}{\sqrt{2^n}} \sum_{\vec{x}} |\vec{x}\,\rangle \right\} \\ &= \sum_{\vec{x}} \frac{1}{\sqrt{2}} \big(|0 \oplus f(\vec{x})\rangle - |1 \oplus f(\vec{x})\rangle\big) \otimes \frac{1}{\sqrt{2^n}} |\vec{x}\,\rangle \\ &= |-\rangle \otimes \frac{1}{\sqrt{2^n}} \sum_{\vec{x}} (-1)^{f(\vec{x})} |\vec{x}\,\rangle, \end{aligned} \tag{5.58}$$

The analog of Eq. (5.47) is

$$H^{\otimes n} |\vec{x}\,\rangle = \frac{1}{\sqrt{2^n}} \sum_{\vec{z}} (-1)^{\vec{x}.\vec{z}} |\vec{z}\rangle, \tag{5.59}$$

where the 'inner product' is computed bitwise modulo 2

$$\vec{x}.\vec{z} = \left( \sum_{j=0}^{2^n-1} x_j z_j \right) \quad \mathrm{mod}\ 2. \tag{5.60}$$

See App. B.4 for a discussion of the inner product for the vector space of bit strings.

Using this identity, the output state of the circuit before the final measurements is

$$|\psi_2\rangle = |-\rangle \otimes \frac{1}{2^n} \sum_{\vec{x}} \sum_{\vec{z}} (-1)^{f(\vec{x})+\vec{x}.\vec{z}} |\vec{z}\rangle. \tag{5.61}$$

By analogy with what we did in the case of the Deutsch algorithm, let us look at the amplitude of the state $|-\rangle \otimes |\vec{z} = \vec{0}\rangle$

$$a_0 = \frac{1}{2^n} \sum_{\vec{x}} (-1)^{f(\vec{x})}. \tag{5.62}$$

If $f$ is constant $a_0 = (-1)^{f(\vec{0})} = \pm 1$ and so all other amplitudes vanish, and the only possible measurement outcome is the all zeroes bit string $\vec{0}$. If $f$ is balanced then $a_0 = 0$ and the all zeroes measurement result *never* occurs. Thus if any of the measurement results is non-zero, we are guaranteed that the function is balanced, and if all the measurement results are zero, we are guaranteed that the function is constant.

Just as for the Deutsch algorithm, a single query of the quantum oracle tells us the global property of the function. Thus the performance of the two quantum algorithms is essentially the same. The difference here lies in the difficulty of the *classical* algorithm to solve these two problems. The classical algorithm requires only two oracle queries to solve the Deutsch problem with certainty. Indeed two queries is enough for Bob to completely specify the function Alice chose, not just whether it is constant or balanced. The Deutsch-Jozsa problem is however much harder since it maps $n$ bits to one bit rather than just 1 bit to 1 bit. To completely learn Alice's function, Bob would have to query the function $2^n$ times, once for each possible argument $\vec{x}$ of the function to learn all of the $f(\vec{x})$ values. How many queries must Bob make to learn only whether the function is constant or balanced? The worst-case classical scenario is when the function is such that Bob queries the oracle $2^n/2$ times (with different arguments) and gets the same result (say 0) every time. This means the function could be constant (if all of the remaining queries return 0) or it could be balanced if all of the remaining queries return 1). Since Alice has given Bob a promise that the function is either constant or balanced, he needs to measure only one more value of the function to be certain of the answer. Thus the so-called 'query complexity' of the best classical algorithm is $2^n/2 + 1$ is exponentially larger than that of the quantum algorithm.

Interestingly, one can use a slightly different measure of the degree of difficulty of the classical algorithm and obtain a very different answer. Suppose that we seek a classical algorithm which is stochastic (i.e., involves randomness in some way) and produces the correct answer with probability $1 - \epsilon$. We can ask how the query complexity grows as we make the acceptable error probability $\epsilon$ smaller and smaller. It turns out that Turing machines (universal classical computers) that contain a source of (true) randomness can create stochastic algorithms that are more powerful than purely deterministic Turing machines. To understand this, suppose that Alice and Bob are adversaries and Alice is trying to choose functions $f$ that make Bob's task as difficult as possible. If she knows that Bob always deterministically orders a certain way (say first querying $f(0000)$, then $f(0001)$, then $f(0010)$, etc. in ascending order), then she can choose functions that produce the worst-case scenario discussed above. Bob however can defeat this strategy if his queries use truly randomly chosen (but non-repeating) arguments $\vec{x}$. If he ever sees any two values of the function that are different from each other, he knows that the function is balanced. If the function is balanced and the arguments $\vec{x}$ are chosen randomly, then any given query result $f(\vec{x})$ is equally likely to be 0 or 1. Suppose Bob is unlucky and after $M$ queries he has seen $f(\vec{x}) = 1$ (say) every time, strongly suggesting to Bob that the function is constant. The probability that a balanced function would give this result (thereby fooling Bob)

$$P_{\text{fail}}(M) = \epsilon = 2 \times 2^{-M}, \tag{5.63}$$

where the extra factor of 2 is from the fact that the first $M$ measurement results could be all 0's or all 1's. This equation is reminiscent of Eq. (4.70) presented in the discussion of cloning and superluminal communication.

Note that Eq. 5.63 assumes that $\vec{x}$ is chosen randomly and allows for the possibility that the same bit string could be chosen more than once. It would be (slightly) better if Bob chose $\vec{x}$ randomly and then removed it from his list so that there would be no repeats. Then the failure probability would be slightly *lower* because he can't be fooled by the same bit strings a second time. Imagine you have a box filled with $N = 2^n$ numbers, half of them being 0 and half of them one. The probability that the first one you draw from the box is a 0 is $\frac{N_0}{N} = \frac{1}{2}$, where $N_0 = N/2$ is the initial number of 0's. After discarding that number, the probability that the second number is also a 0 is $\frac{N_0-1}{N} = \frac{1}{2} - \frac{1}{N}$. Continuing this process we see that the failure probability in Eq. 5.63 should replaced by

$$\tilde{P}_{\text{fail}}(M) = 2 \times 2^{-M} \prod_{j=0}^{M-1} \left(1 - \frac{2j}{N}\right), \tag{5.64}$$

$$\approx 2 \times 2^{-M} \prod_{j=0}^{M-1} e^{-\frac{2j}{N}} \approx 2 \times 2^{-M} e^{-\frac{M(M-1)}{N}}. \tag{5.65}$$

where we have made the reasonable assumption that $M \ll N$ since we do not need to consider large $M$ to drive the failure probability extremely low. The rule of not repeating any queries with the same $\vec{x}$ makes the failure probability (slightly) smaller.

These results show us that there is a huge reduction in query complexity if you are willing to accept a tiny (exponentially small) chance of failure. Another way to say it is that the average case is much easier classically than the worst case.

**Exercise 5.3.** Show that the number of balanced functions

$$f : \{0,1\}^n \to \{0,1\}$$

is given by the binomial coefficient

$$\left( \begin{array}{c} 2^n \\ 2^{n-1} \end{array} \right).$$

Using the Stirling (asymptotic) approximation for the factorial of a number,

$$\ln M! \sim (M + \frac{1}{2}) \ln M - M + \frac{1}{2} \ln(2\pi) + \ldots,$$

show that the fraction $F(n)$ of all functions that are balanced rapidly becomes small as $n$ increases, and asymptotically approaches

$$F(n) \sim \frac{2}{\sqrt{2\pi 2^n}}.$$

An intuitive way to think about this result is the following. Rather than considering all possible functions $f$, consider a single example of $f$ whose value $f(\vec{x}) = \pm 1$ is chosen randomly for each value of its argument $\vec{x}$. Then the probability distribution for $S = \sum_{\vec{x}} f(\vec{x})$ will, to a good approximation, be a Gaussian with mean $N/2$ and variance $N/4$ with $N = 2^n$

$$P(S) \approx \sqrt{\frac{1}{2\pi(N/4)}} e^{-\frac{1}{2(N/4)}(S-N/2)^2}.$$

Note that for $N \gg 1$ the Gaussian is such a slowly varying function that we can replace the normalization condition $\sum_{S=0}^{N} P(S)$ by $\int_{-\infty}^{+\infty} dS\, P(S)$. The probability of the function being balanced is

$$F(n) = P(N/2) \approx \frac{2}{\sqrt{2\pi 2^n}}.$$

## 5.6 Bernstein-Vazirani Problem

[The discussion here follows Riefel and Polak, Sec. 7.5.2.]

The Bernstein-Vazirani problem bears some relationship to the Deutsch-Jozsa problem that we studied previously and is defined in the following way.

Alice creates a function $f : \{0,1\}^{\otimes n} \to \{0,1\}$ of the form

$$f(\vec{x}) = \vec{u}.\vec{x}, \tag{5.66}$$

where

$$\vec{u} = (u_{n-1}u_{n-2}\dots u_2u_1u_0) \tag{5.67}$$

is a fixed binary vector that Alice chooses. Bob's task is to learn $\vec{u}$ with the smallest possible number of queries of the function $f$. Before studying the quantum algorithm, let us think about the query complexity in the classical case. The vector $\vec{u}$ is unknown to Bob since Alice has chosen it. Bob can query the function with a sequence of input vectors each of which contains only a single non-zero entry

$$\begin{aligned}
\vec{x}_0 &= (00\dots001)\\
\vec{x}_1 &= (00\dots010)\\
\vec{x}_2 &= (00\dots100)\\
&\dots\\
\vec{x}_{n-2} &= (01\dots000) && (5.68)\\
\vec{x}_{n-1} &= (10\dots000), && (5.69)
\end{aligned}$$

and with each query will learn one bit in the string $\vec{u}$ because

$$f(\vec{x}_j) = u_j. \tag{5.70}$$

Since Alice could have chosen the bits in the string $\vec{u}$ at random, there is no shortcut that can reduce the classical query complexity below $n$, the length of the bit string.

Remarkably, the Bernstein-Vazirani quantum algorithm can find $\vec{u}$ (with certainty) with only a single query to the quantum oracle that encodes the function $f$! The circuit is the same one used for the Deutsch-Jozsa algorithm and the output is given by Eq. (5.61). However because of the particular form of the function given in Eq. (5.66), we can say more about the solution

$$\begin{aligned}|\psi_2\rangle &= |-\rangle \otimes \frac{1}{2^n}\sum_{\vec{x}}\sum_{\vec{z}}(-1)^{f(\vec{x})+\vec{x}.\vec{z}}|\vec{z}\rangle \\ &= |-\rangle \otimes \frac{1}{2^n}\sum_{\vec{x}}\sum_{\vec{z}}(-1)^{\vec{u}.\vec{x}+\vec{x}.\vec{z}}|\vec{z}\rangle \\ &= |-\rangle \otimes \frac{1}{2^n}\sum_{\vec{x}}\sum_{\vec{z}}(-1)^{(\vec{u}\oplus\vec{z}).\vec{x}}|\vec{z}\rangle \\ &= |-\rangle \otimes |\vec{u}\rangle. \end{aligned} \tag{5.71}$$

The last line follows from the fact that if $\vec{y} = \vec{u} \oplus \vec{z} \neq \vec{0}$ then the function $g(\vec{x}) = \vec{y}.\vec{x}$ is necessarily balanced (see Box 5.1), meaning that

$$\sum_{\vec{x}}(-1)^{g(\vec{x})} = 0 \tag{5.72}$$

for all $\vec{y} \neq \vec{0}$. Because $\vec{y} = \vec{u} \oplus \vec{z} = \vec{0} \iff \vec{z} = \vec{u}$, the only term that survives the sum over $\vec{x}$ is $\vec{z} = \vec{u}$. Thus the output measurements from the circuit immediately give us the unknown vector $\vec{u}$ after only a single call to the oracle thereby providing a polynomial (in $n$) quantum advantage.

David Mermin provided an interesting alternative view of the Bernstein-Vazirani circuit based on constructing the actual quantum circuit shown in Fig. 5.9 that realizes the oracle. Using this oracle with the $b$ register initialized in $|+\rangle^{\otimes n}$ and $d_0$ initialized in $|-\rangle$ as shown in the left panel of Fig. 5.10 executes the Bernstein-Vazirani algorithm analyzed above. We can gain a new understanding of the algorithm by inserting identity operators in the form of pairs of Hadamard gates on each side of each CNOT gate. Using the identity from Eq. (5.6) illustrated in the lower panel of Fig. 5.2 yields the equivalent circuit shown in the right panel of Fig. 5.10. Now we see immediately that because $d_0$ is in state $|1\rangle$, the algorithm works by simply mapping the $b$ register from its input value $\vec{b} = \vec{0}$ to the output value $\vec{b} = \vec{u}$. No quantum superpositions appear anywhere in this version of the algorithm! Alice has tried to challenge Bob by hiding a series of CNOTs inside the oracle that are controlled by a random (unknown to Bob) subset of the input qubits and that all target the output qubit. It seems like it would be impossible for Bob to meet Alice's challenge in a single query of the oracle. However Bob can cleverly turn the tables on Alice by working in the $X$ basis, interchanging the control and target qubits, thereby giving himself control of the situation and flipping only those input bits $j$ for which $u_j = 1$, thereby decoding Alice's hidden bit string in a single query of the oracle!

**Box 5.1. Balanced Function Lemma**
Let $\vec{x}$ and $\vec{y}$ be bit strings of length $n$. Define the function $g : \{0,1\}^n \to \{0,1\}$ via

$$g(\vec{x}) = \vec{y}.\vec{x} = \left(\sum_{j=0}^{n-1} y_j x_j\right) \mod 2. \tag{5.73}$$

**Lemma**: $g(\vec{x})$ is a balanced function (outputs 0 and 1 an equally often) provided $\vec{y} \neq \vec{0}$.
**Proof**: Since $\vec{y} \neq \vec{0}$, $\vec{y}$ has at least one component that is non-zero (i.e. is unity). Let the index of the first (counting from right to left) non-zero component of $\vec{y}$ be $k$. Then every vector $\vec{x}$ has a unique partner $\vec{x}'$ which differs from $\vec{x}$ only at position $k$. It follows immediately that

$$\begin{aligned} g(\vec{x}) + g(\vec{x}') &= 1 \\ (-1)^{g(\vec{x})} + (-1)^{g(\vec{x}')} &= 0 \\ \sum_{\vec{x}} (-1)^{g(\vec{x})} &= 0, \end{aligned} \tag{5.74}$$

and thus $g(\vec{x})$ is balanced. Boxes 5.2 and 5.3 present more complex but interesting and informative alternative proofs.

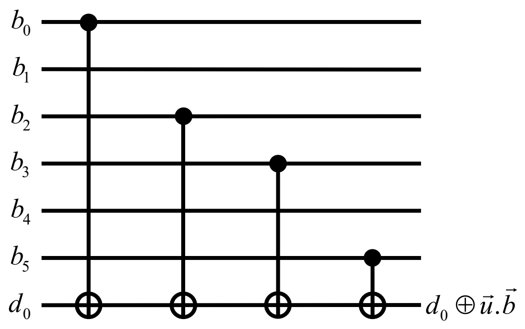

Figure 5.9: Circuit realization of the Bernstein-Vazirani oracle for the function $f(\vec{b}) = \vec{u}.\vec{b}$ with (in this case) $\vec{u} = (u_5u_4u_3u_2u_1u_0) = (101101)$. The CNOT gates all have $d_0$ as their target, but the controls are limited to those wires $j$ for which $u_j = 1$. Thus the CNOT will flip $d_0$ only if $b_ju_j = 1$ and the net result after all the CNOTs have been applied is $d_0 \to d_0 \oplus \vec{u}.\vec{b}$.

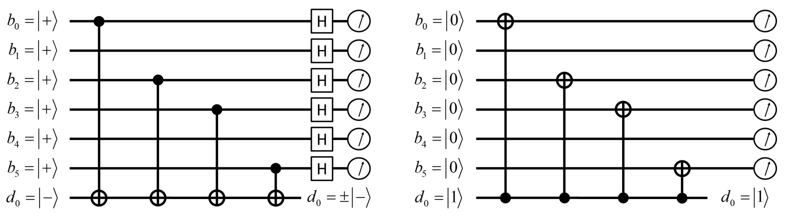

Figure 5.10: Left panel: Circuit realizing the Bernstein-Vazirani oracle and algorithm for the function $f(\vec{b}) = \vec{u}.\vec{b}$ with (in this case) $\vec{u} = (u_5u_4u_3u_2u_1u_0) = (101101)$. The CNOT gates all have $d_0$ as their target, but the controls are limited to those wires $j$ for which $u_j = 1$. Thus the CNOT will flip $d_0$ only if $b_ju_j = 1$ and the net result after all the CNOTs have been applied is $d_0 \to d_0 \oplus \vec{u}.\vec{b}$. Right panel: Mermin's analysis of the same circuit obtained by conjugating each CNOT with identity operations $H^2 = I$. The controls are now all $d_0 = 1$ and the targets are those input lines $j$ for which $u_j = 1$. Thus the output register maps to $\vec{b} = \vec{u}$.

**Box 5.2. Balanced Function Lemma Alternative Proof I:** Let $\vec{x}$ and $\vec{y}$ be bit strings of length $n$. Define the function $g : \{0,1\}^n \to \{0,1\}$ via

$$g(\vec{x}) = \vec{y}.\vec{x} = \left(\sum_{j=0}^{n-1} y_j x_j\right) \mod 2. \tag{5.75}$$

**Lemma**: $g(\vec{x})$ is a balanced function provided $\vec{y} \neq \vec{0}$.
**Proof**: Note that the following summation over all bit strings $\vec{x}$ contains $2^n$ terms (and that for this particular calculation we can drop the modular arithmetic)

$$Q \equiv \sum_{\vec{x}} (-1)^{g(\vec{x})} = \sum_{x_0=0,1} \cdots \sum_{x_{n-1}=0,1} (-1)^{\sum_j x_j y_j} = \prod_{j=0}^{n-1} \left( \sum_{x_j=0,1} (-1)^{x_j y_j} \right) \tag{5.76}$$

Let $S$ be the set of indices $j$ for which $y_j = 1$. Since $\vec{y} \neq \vec{0}$, this set is not empty. Let $\bar{S}$ be its complement (i.e., the set of indices $k$ for which $y_k = 0$). Then we can rewrite the product above as two products

$$Q = \prod_{k \in \bar{S}} \left( \sum_{x_k=0,1} (-1)^0 \right) \prod_{j \in S} \left( \sum_{x_j=0,1} (-1)^{x_j} \right). \tag{5.77}$$

Each term in the second product vanishes. Hence $Q = 0$ and thus $g$ is balanced. QED.

**Box 5.3. Balanced Function Lemma Alterantive Proof II:** Let $\vec{x}$ and $\vec{y}$ be bit strings of length $n$. Define the function $g : \{0,1\}^n \to \{0,1\}$ via

$$g(\vec{x}) = \vec{y}.\vec{x}. \tag{5.78}$$

**Lemma**: $g(\vec{x})$ is a balanced function provided $\vec{y} \neq \vec{0}$.
**Proof**: note that the following summation over all possible bit strings $\vec{x}$ contains $2^n$ terms

$$g(\vec{x}) = \left(\sum_{j=0}^{n-1} y_j x_j\right) \mod 2 \quad = \quad m \mod 2, \text{ where} \tag{5.79}$$

$$m \equiv \sum_{j\in S} x_j, \tag{5.80}$$

and the summation in the last line is only over the set $S$ of $j$ values for which $y_j = 1$. Since $\vec{y} \neq \vec{0}$, this set is not empty. Let $k$ be the cardinality of the set $S$ (i.e., the number of non-zero bits in $\vec{y}$). The complement of this set, $\bar{S}$ has cardinality $n-k$ (i.e., the number of zero bits in $\vec{y}$ is $n-k$). The value of $m$ can range from 0 (if for all the $j \in S$, $x_j = 0$) to $k$ (if for all the $j \in S$, $x_j = 1$). The values of $x_j$ for $j \in \bar{S}$ do not contribute since the corresponding $y_j$'s vanish. Using these facts we can write

$$\sum_{\vec{x}} (-1)^{g(\vec{x})} = \sum_{m=0}^{k} G_m (-1)^m, \tag{5.81}$$

where $G_m$ is the number of times that a given value of $m$ occurs when summing over $\vec{x}$

$$G_m = 2^{n-k} \begin{pmatrix} k \\ m \end{pmatrix}. \tag{5.82}$$

The power of 2 is the contribution from the sum in the LHS of Eq. (5.81) over the $(n-k)$ components of $\vec{x}$ in $\bar{S}$, and the binomial factor comes from the sum over the $k$ components of $\vec{x}$ in $S$ subject to the constraint in Eq. (5.80); i.e., it is the number of distinct ways of distributing $m$ 1's and $(k-m)$ 0's among the $k$ positions within the set $S$. Finally we obtain

$$\sum_{\vec{x}} (-1)^{g(\vec{x})} = 2^{n-k} \sum_{m=0}^{k} (+1)^{k-m} (-1)^m \begin{pmatrix} k \\ m \end{pmatrix} = 2^{n-k}[1 + (-1)]^k = 0. \tag{5.83}$$

Hence $g$ is balanced. QED.

## 5.7 Simon's Algorithm

Simon's algorithm solves another toy problem but, as noted in Box 5.5, has (unlike Deutsch-Jozsa) the distinction of providing *exponential* oracle separation from BPP. While it only solves a toy problem, Simon's algorithm inspired important algorithms based on the quantum Fourier transform and Shor's factoring algorithm.

The problem statement is the following:

1. Given a binary function $\vec{f} : \{0,1\}^n \to \{0,1\}^n$ and
2. Given a guarantee that $\vec{f}$ is either one-to-one or two-to-one, and
3. Given a further guarantee that
$$\vec{f}(\vec{x}_1) = \vec{f}(\vec{x}_2) \iff \vec{x}_2 = \vec{x}_1 \oplus \vec{b}, \text{ for fixed } \vec{b},$$
4. Task: find the 'hidden' bit string $\vec{b}$. (Note that $\vec{f}$ is one-to-one $\iff$ $\vec{b} = \vec{0}$.)

We write $\vec{f}$ as a vector-valued function because it maps the set of binary vectors of length $n$ into itself. Table 5.2 gives three different examples of functions for the case $n = 2$ and $\vec{b} = (1, 0)$.

| $\vec{x}$ | $\vec{x} \oplus \vec{b}$ | $\vec{f}_1(\vec{x})$ | $\vec{f}_2(\vec{x})$ | $\vec{f}_3(\vec{x})$ |
|---|---|---|---|---|
| 00 | 10 | 00 | 10 | 01 |
| 01 | 11 | 11 | 01 | 11 |
| 10 | 00 | 00 | 10 | 01 |
| 11 | 01 | 11 | 01 | 11 |

Table 5.2: Three example functions obeying the guarantees required for Simon's problem with $n = 2$ and $\vec{b} = (1, 0)$.

### Classical Query Complexity for Simon's Problem

Let us begin our analysis of Simon's problem by considering how hard it is to solve classically. That is, what is the query complexity? Suppose that (somehow) we discover two inputs with the same output

$$\vec{f}(\vec{x}_1) = \vec{f}(\vec{x}_2). \tag{5.84}$$

Then we know that the function is two-to-one and we can easily solve for the unknown $\vec{b}$ using the identity

$$\begin{aligned}\vec{x}_1 \oplus \vec{x}_2 &= \vec{x}_1 \oplus (\vec{x}_1 \oplus \vec{b}) \\ &= (\vec{x}_1 \oplus \vec{x}_1) \oplus \vec{b} = \vec{b}.\end{aligned} \tag{5.85}$$

Thus is we can find a matching pair of inputs, we have solved the problem. Note that if we *cannot* find a matching pair, then we have shown that the function is one-to-one and have also solved the problem (since $\vec{b} = \vec{0}$ in this case).

How hard is it to find a pair with matching outputs? In the worst case, to be absolutely sure that there are no matching pairs we would need to evaluate $N = 2^n/2 + 1$ inputs. This is because if we evaluate only half of the possible inputs, the partners of all those inputs might accidentally be in the other half. Hence the query complexity for a deterministic solution is exponential in $n$, just as for the Deutsch-Jozsa problem.

As we did for Deutsch-Jozsa, let us now ask what the query complexity is for a probabilistic algorithm. Suppose we query $\vec{f}$ $M$ times with $M$ distinct inputs $\vec{x}$. The number of distinct pairs in our sample is

$$M_{\text{pairs}} = \frac{M(M-1)}{2}, \tag{5.86}$$

where the factor of 2 in the denominator is to prevent counting $\{\vec{x}_j, \vec{x}_k\}$ and $\{\vec{x}_k, \vec{x}_j\}$ as different pairs.

Given some $\vec{x}_1$, and assuming $\vec{f}$ is two-to-one, the probability that a randomly selected $\vec{x}_2 \neq \vec{x}_1$ yields $\vec{f}(\vec{x}_2) = \vec{f}(x_1)$ is

$$\epsilon = \frac{1}{2^n - 1} \sim 2^{-n}. \tag{5.87}$$

This is because there are $2^n - 1$ choices for $x_2$ and only one of them gives a matching output (assuming $\vec{f}$ is two-to-one). There is only one way to fail to find a match in a sample of size $M$: no pairs match. This occurs with probability

$$\begin{aligned}P_{\text{fail}} &= (1-\epsilon)^{\frac{M(M-1)}{2}} \\ &= e^{\ln(1-\epsilon)\frac{M(M-1)}{2}} \approx e^{-\epsilon\frac{M(M-1)}{2}},\end{aligned} \tag{5.88}$$

where the last (approximate) equality follows from Taylor series expanding the logarithm to first order in $\epsilon$. Thus if $\epsilon\frac{M(M-1)}{2} > 1$, there is a reasonable chance of success (which rapidly approaches unity as $M$ exceeds this threshold). For small $\epsilon$, the threshold value for $M$ is

$$\begin{aligned} M_0 &\sim \left(\frac{2}{\epsilon}\right)^{\frac{1}{2}} \\ &\sim \left(\frac{2}{2^{-n}}\right)^{\frac{1}{2}} \\ &\sim 2^{\frac{n+1}{2}}. \end{aligned} \tag{5.89}$$

We see that because though the number of pairs in our samples scales quadratically with the sample size,[1] we can begin to be reasonably sure of finding a matching pair after sampling only about $\sqrt{2^n}$ times, even though the worst case scenario would require sampling $2^{(n-1)} + 1$ times. Nevertheless, we still require a number of samples that is exponential in $n$ to insure a reasonable probability of success. Thus, unlike the Deutsch-Jozsa problem, Simon's problem lies outside complexity class BPP. we can use a probabilistic algorithm to obtain a bounded error, but not in polynomial time. Thus Simon's problem is exponentially (in $n$) harder than Deutsch-Jozsa for a classical computer, even if we allow probabilistic computation.

## Quantum Query Complexity for Simon's Problem

Let us now consider Simon's quantum algorithm which shows that the problem is in BQP (see Box 5.5) thereby giving exponential quantum advantage. We will use the quantum circuit shown in Fig. 5.11 where $U_f$ is a quantum oracle that encodes the function $f$ and is of the form shown in Fig. 5.7. For explanatory purposes, we will initially take the case in which the function $f$ is two-to-one and not one-to-one.

The Hadamard gates put the upper register in a uniform superposition of all its possible states so that the state of the system immediately after the

[1]This quadratic scaling of the number of distinct pairs is the origin of the so-called 'birthday paradox.' In a group of $M$ people the probability of two people having the same birthday is larger than you might naively guess.

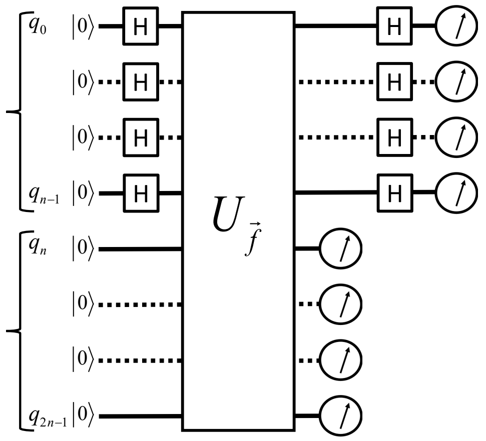

Figure 5.11: Circuit for Simon's algorithm.

application of the oracle is

$$|\psi_1\rangle = \frac{1}{\sqrt{2^n}} \sum_{\vec{x}} \left|\vec{f}(\vec{x})\right\rangle \otimes \left|\vec{x}\,\right\rangle. \tag{5.90}$$

It is convenient at this point to partition the set $S$ of all the vectors $\vec{x}$ into two disjoint parts parts $S = S_1 \cup S_2$ defined by $\vec{x} \in S_1 \iff (\vec{x} \oplus \vec{b}) \in S_2$. That is each $\vec{x}$ and its partner $\vec{x} \oplus \vec{b}$ are in opposite subsets. It does not matter which subset we choose to put $\vec{x}$ in, only that its partner is not in the same subset. With this partition, Eq. (5.90) can be written

$$|\psi_1\rangle = \frac{1}{\sqrt{2^{(n-1)}}} \sum_{\vec{x} \in S_1} \left|\vec{f}(\vec{x})\right\rangle \otimes \frac{1}{\sqrt{2}} \left[\left|\vec{x}\,\right\rangle + \left|\vec{x} \oplus \vec{b}\right\rangle\right], \tag{5.91}$$

where we have taken advantage of the fact that $\vec{f}(\vec{x} \oplus \vec{b}) = \vec{f}(\vec{x})$. It is important to understand that the bit string $\vec{f}(\vec{x})$ is unique in the sense that the value of $\vec{f}(\vec{x})$ is unique for every vector $\vec{x} \in S_1$. The only time the same bit string occurs is for the partner $\vec{x} \oplus \vec{b}$ which lies in $S_1$.

As shown in Fig. 5.11, the next step is to measure the lower register. It turns out that this step is not actually necessary, but it greatly simplifies the analysis. Measurement of the lower register tells us the unique vector $\vec{f}(\vec{y})$ for some random (and as yet unknown) vector $\vec{y} \in S_1$. As a result the state collapses to

$$|\psi'\rangle = \frac{1}{\sqrt{2}}|\vec{f}(\vec{y})\rangle \otimes \left\{|\vec{y}\,\rangle + |\vec{y}\oplus\vec{b}\,\rangle\right\}. \tag{5.92}$$

Crucially, because the function (we are assuming) is two-to-one, the upper register only partially collapses, ending up in a superposition of the two states that are consistent with the measurement result because they yield the same value of the function (same state within the register being measured). (For a reminder about partial state collapse under measurement of a subset of the qubits in a system, see Sec. 4.2.) At this point we have learned the value of $\vec{f}$ but not the value of $\vec{y}$ or $\vec{y}\oplus\vec{b}$.

Stop for a moment and consider just how powerful this result is. The huge superposition state has collapsed to a very small superposition that automatically picked out (at random) a state $|\vec{y}\,\rangle$ and its partner $|\vec{y}\oplus\vec{b}\rangle$! This is a huge advantage–but we are not yet home free. If we measure the upper register, the state collapses to *either* $|\vec{y}\,\rangle$ *or* $|\vec{y}\oplus\vec{b}\rangle$ and we have lost all the information we needed to find $\vec{b}$ using $\vec{b} = \vec{y}\oplus(\vec{y}\oplus\vec{b})$.

The solution to this difficulty is shown in Fig. 5.11. Before measuring the upper register we apply $H^{\otimes n}$ to it to obtain a kind of quantum interference between $|\vec{y}\,\rangle$ and $|\vec{y}\oplus\vec{b}\rangle$ that will allow us to determine $\vec{b}$

$$|\psi''\rangle = \frac{1}{\sqrt{2}}|\vec{f}(\vec{y})\rangle \otimes H^{\otimes n}\left\{|\vec{y}\rangle + |\vec{y}\oplus\vec{b}\rangle\right\}. \tag{5.93}$$

Recalling Eqs. (5.47,5.59), and Eq. (5.60) we obtain

$$|\psi''\rangle = \frac{1}{\sqrt{2}}|\vec{f}(\vec{y})\rangle \otimes \frac{1}{\sqrt{2^n}}\sum_{\vec{z}}(-1)^{\vec{y}.\vec{z}}\left\{1+(-1)^{\vec{b}.\vec{z}}\right\}|\vec{z}\rangle. \tag{5.94}$$

Noting that the bit-string vector dot product is computed modulo 2, we see immediately that if $\vec{b}.\vec{z} = 1$, the term in curly brackets vanishes, while if $\vec{b}.\vec{z} = 0$, then the term in curly brackets is $+2$. We conclude therefore that the measurement collapses the state with equal probabilities onto all possible states $|\vec{z}\rangle$ obeying

$$\vec{b}.\vec{z} = 0. \tag{5.95}$$

Now if $\vec{b} \neq \vec{0}$ is a vector of length $n$ then from the Balanced Function Lemma in Box 5.1, we know that there are $2^{n-1}$ different solutions to this equation, comprising one half of the set of all possible vectors. The measurement results yield random vectors $\vec{z}$ of length $n$ that are 'perpendicular' to $\vec{b}$. If we run the algorithm $n-1+m$ times with $m$ being a small integer constant of order unity, it is highly likely that measurement results $\{\vec{z}_j; j \in [1, n-1+m]\}$ will yield a set of $n-1$ linearly independent non-zero vectors $\vec{v}_j; j \in 1, n-1$, so that the set of linear equations

$$\begin{aligned}
\vec{b}.\vec{v}_1 &= 0 \\
\vec{b}.\vec{v}_2 &= 0 \\
\vec{b}.\vec{v}_3 &= 0 \\
&\vdots \\
\vec{b}.\vec{v}_{n-1} &= 0
\end{aligned} \tag{5.96}$$

can be uniquely solved for $\vec{b}$ using Gaussian elimination. As a simple example, consider the case $n = 2$ and $\vec{b} = (01)$. There is exactly $n - 1 = 1$ non-zero solution of Eq. (5.95), namely $\vec{z} = (10)$. Note that if we find that the algorithm produces $n$ rather than $n-1$ linearly independent non-zero vectors, then we know that $\vec{b} = \vec{0}$ and the function $f$ is one-to-one. (The dimension of the vector space is $n$ so the largest set of linearly independent vectors is $n$.) Interestingly, if $\vec{b}$ contains an even number of non-zero entries (e.g., $\vec{b} = (11)$) then $\vec{b}.\vec{b} = 0$ and $\vec{b}$ itself is one of the solutions (in the $n = 2$ example, the only non-zero solution).

As we will discuss shortly, it turns out that the failure probability falls exponentially with $m$, the number of queries beyond $n - 1$. Thus $m$ does not need to increase as $n$ increases. Thus the quantum algorithm is probabilistic (it may fail to find enough linearly independent vectors for any fixed number of queries, $n - 1 + m$), but has a query complexity polynomial (in this case linear) in $n$, whereas the best classical algorithm is exponential in $n$. The quantum complexity class (see Box 5.5) is thus BQP (Bounded Error Quantum Polynomial Time).

## Failure Probability of Simon's Algorithm for fixed query number

For simplicity we will continue to assume that $\vec{b} \neq \vec{0}$ so that there are only $n - 1$ linearly independent vectors satisfying Eq. (5.95). If $\vec{b} = \vec{0}$, there will be $n$ such vectors and the discussion below is easily modified for this case by replacing $n - 1$ with $n$ in the various formulae we will discuss.

As mentioned above Simon's algorithm outputs a single bit string $\vec{z}$ of length $n$ obeying Eq. (5.95). There are $2^{n-1}$ such bit strings, all equally likely. The only way to fail is if $\vec{z}_1 = \vec{0}$. Hence the probability of obtaining a non-zero result on the first try is very close to unity for large $n$

$$P = 1 - 2^{-(n-1)}. \tag{5.97}$$

Next suppose that we have obtained a non-zero $\vec{v}_1 = \vec{z}_1$. There are now two ways to fail to advance when we call the oracle a second time: either $\vec{z}_2 = \vec{0}$ or $\vec{z}_2 = \vec{z}_1$. Hence the success probability to obtain a second vector $\vec{v}_2$ that is non-zero and is linearly independent of $\vec{v}_1$ is

$$P = 1 - 2^{-(n-2)}. \tag{5.98}$$

What happens when we continue to iterate this process? Suppose that we are at the point where we have found $\ell$ linearly independent solutions $\vec{v}_1, \ldots, \vec{v}_\ell$. These span a subspace of dimension $2^\ell$ that includes $\vec{0}$. The next step of the algorithm will fail to produce a new (non-zero) linearly independent vector $\vec{v}_{\ell+1}$ if the algorithm outputs a vector in this subspace. Hence the failure probability is

$$P_{\ell\to\ell} = \frac{2^\ell}{2^{(n-1)}} = 2^{\ell-(n-1)}, \tag{5.99}$$

and the success probability to transition from having $\ell$ to $\ell + 1$ linearly independent non-zero vectors is

$$P_{\ell\to\ell+1} = \frac{2^{n-1} - 2^\ell}{2^{n-1}} = 1 - 2^{\ell-(n-1)}. \tag{5.100}$$

Notice that this expression for $P_{0\to1}$ agrees with Eq. (5.97) and this expression for $P_{1\to2}$ agrees with Eq. (5.98). The final transition to the goal has the lowest success probability: $P_{n-2\to n-1} = \frac{1}{2}$. Also notice that $P_{(n-1)\to n} = 0$ (so that $P_{(n-1)\to(n-1)} = 1$) as required since the number of linearly independent vectors cannot exceed $n - 1$ (for the case $\vec{b} \neq \vec{0}$) and the transitions must terminate.

These transition probabilities define a Markov chain illustrated in Fig. 5.12. The probability of passing through the chain from $\ell = 0$ to $\ell = (n-1)$ in $n-1$ steps (i.e., successfully advancing each time with zero failures) is

$$\mathcal{P}_0(n) = \prod_{\ell=0}^{n-2} P_{\ell\to\ell+1} = \prod_{\ell=0}^{n-2}[1 - 2^{\ell-(n-1)}]. \tag{5.101}$$

In the limit of large $n$ the success probability in the initial stages is extremely close to unity. As a result the probability of reaching the goal with zero failures approaches a constant given by the infinite product

$$\mathcal{P}_0(\infty) = \left(\frac{1}{2}\right)\left(\frac{3}{4}\right)\left(\frac{7}{8}\right)\left(\frac{15}{16}\right)\left(\frac{31}{32}\right)\dots \approx 0.288788095. \tag{5.102}$$

Every 'trajectory' must pass through the chain from beginning to end and so the factor $\mathcal{P}_0(n)$ appears in the probability of every trajectory.

Let us now consider all the trajectories that involve precisely one failure (an hence require $n-1+1=n$ calls to the oracle). The failure to advance can occur at any site from $\ell=0$ to $\ell=n-2$. Summing over all these distinct trajectories gives

$$\mathcal{P}_1(n) = \mathcal{P}_0(n)\sum_{\ell=0}^{n-2} P_{\ell\to\ell} = \mathcal{P}_0(n)S_1(n), \tag{5.103}$$

where

$$S_1(n) \equiv \sum_{\ell=n-2}^{0} 2^{\ell-(n-1)} \tag{5.104}$$

$$= \frac{1}{2} + \frac{1}{4} + \frac{1}{8} + \dots \frac{1}{2^{n-1}}, \tag{5.105}$$

where for convenience we have reversed the order of the summation. In the limit of large $n$ this approaches

$$S_1(\infty) = \frac{1}{2}\left(\frac{1}{1-\frac{1}{2}}\right) = 1, \tag{5.106}$$

which yields a total probability of success in $n$ steps or fewer which is greater than 50%

$$\mathcal{P}_0(\infty) + \mathcal{P}_1(\infty) = 2\mathcal{P}_0(\infty) \approx 0.57757619. \tag{5.107}$$

A little thought shows that the corresponding sum over all possible combinations of two failure probabilities is

$$S_2(\infty) = \frac{1}{2}\left[\sum_{\ell=0}^{\infty} P_{\ell\to\ell}\right]^2 + \frac{1}{2}\sum_{\ell=0}^{\infty}[P_{\ell\to\ell}]^2 = \frac{1}{2}\left[1+\frac{1}{3}\right] = \frac{2}{3}. \tag{5.108}$$

From this we obtain the total probability to arrive at the goal in at most $n+1$ calls to the oracle is

$$\mathcal{P}_0(\infty) + \mathcal{P}_1(\infty) + \mathcal{P}_2(\infty) = \left[2+\frac{2}{3}\right]\mathcal{P}_0(\infty) \approx 0.770102. \tag{5.109}$$

We thus see that the probability is rapidly converging towards unity as the number of calls $n-1+m$ to the oracle increases beyond the minimum $n-1$. With further effort it is possible show that this convergence is exponential in $m$. Roughly speaking, the last step from $\ell = n-2$ to $\ell = n-1$ is the bottleneck in the dynamics since it has the lowest success probability $P_{n-2\to n-1} = \frac{1}{2}$. From this it follows that the probability to fail to reach the goal for large $m$ is proportional to $2^{-m}$.

**Box 5.4. Exact Markov Chain Analysis** [STILL UNDER CONSTRUCTION] We can generalize the above to obtain a formally exact expression for the success probability for trajectories involving $m$ failures to advance

$$\mathcal{P}_m(n) = \mathcal{P}_0(n)\sum_{\vec{\mu}} G(\vec{\mu},m)\prod_{j=0}^{n-2} P_{j\to j+1}^{\mu_j}, \tag{5.110}$$

where $\vec{\mu}$ is a vector of length $n$ whose entries are non-negative integers representing the number of failures on each site of the Markov chain, and the constraint that there be precisely $m$ failures is enforced by

$$\begin{aligned} G(\vec{\mu},m) = 1 &\iff \sum_{j=0}^{n-1}\mu_j = m \\ &= 0 \ \text{otherwise}. \end{aligned} \tag{5.111}$$

[STILL UNDER CONSTRUCTION.]

One relatively simple quantity to compute is the mean number of failures to advance at each node of the Markov chain. For node $\ell$, the probability of $k$ failures before advancing is $(P_{\ell\ell})^k$. Hence the mean number of failures is

$$\bar{f}_\ell = \sum_{k=0}^{\infty} k(P_{\ell\ell})^k. \tag{5.112}$$

This sum can be evaluated by defining the generating function

$$f_\ell(\lambda) = \sum_{k=0}^{\infty} e^{-\lambda k} P_{\ell\ell}^k = \frac{1}{1 - e^{-\lambda} P_{\ell\ell}}, \tag{5.113}$$

and noting that

$$\bar{f}_\ell = -\frac{d}{d\lambda} f_\ell(\lambda)\bigg|_{\lambda=0} = \frac{P_{\ell\ell}}{(1 - P_{\ell\ell})^2}. \tag{5.114}$$

The mean number of failures before reaching the end of the Markov chain is thus

$$\begin{aligned} \bar{m}(n) &= \sum_{\ell=0}^{n-2} \frac{P_{\ell\ell}}{(1 - P_{\ell\ell})^2} \\ &= \sum_{k=1}^{n-1} \frac{2^{-k}}{(1 - 2^{-k})^2} \approx 2.744, \end{aligned} \tag{5.115}$$

where the last equality is the result for asymptotically large $n$ but is accurate (to three digits after the decimal point) for any $n \geq 16$. The small value of the mean number of failures is consistent with the success probability rapidly approaching unity for $m \sim 3$ and larger. [Note this argument does not rule out the possibility of a long tail in the distribution of the number of failures. However a similar calculation of the mean square number of failures does rule this out. Appendix G in Mermin, Quantum Computer Science, provides a lower bound on the success probability after $n - 1 + m$ queries.]

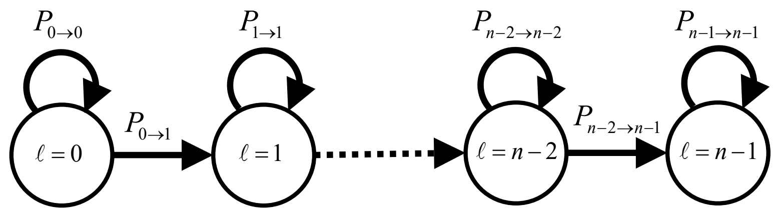

Figure 5.12: Markov chain showing the state transition probabilities of Simon's algorithm as one calls the oracle in an attempt to increase the number of linearly independent output vectors from $\ell$ to $\ell+1$. $P_{\ell\to\ell}$ is the probability of failing to advance and $P_{\ell\to\ell+1}$ is the success probability.

**Box 5.5. Complexity Classes** Theoretical computer science devotes much effort to defining different complexity classes that classify the degree of difficulty of solving different types of problems within various different computational models. Scott Aaronson and collaborators maintain a 'Complexity Class Zoo' at https://complexityzoo.net/Complexity_Zoo.
The Deutsch-Jozsa problem is an oracle problem in the complexity class known as EQP, meaning that it can be solved *exactly* in polynomial time (oracle query complexity is polynomial in the number of bits in the input to the function $f$). In fact the time is a constant independent of $n$. This is exponentially better than the best classical algorithm that exactly solves the problem which we saw requires a number of oracle queries $2^n/2+1$ which is exponential in $n$.
However we also saw that there exists a fast probabilistic classical algorithm that solves the problem using only a polynomial number of queries $N$ provided that the user is willing to accept an exponentially small failure probability $\epsilon = 22^{-N}$ (that is independent of $n$). This complexity class is called BPP, meaning that problems in this class can be solved on a classical computer with bounded error in polynomial time.
In short, the Deutsch-Jozsa quantum algorithm is not better than the best classical algorithm if one is willing to allow probabilistic classical algorithms and thus accept an exponentially small failure probability. In this case 'not better' means that the quantum and probabilistic classical algorithms only need a fixed number of oracle queries, independent of the problem size. (The number of queries required for the probabilistic classical algorithm depends on the logarithm of $1/\epsilon$.)
As we will see, the Berstein-Vazirani algorithm studied in Sec. 5.6 provides an 'oracle separation' between complexity classes (see Box 5.5) BPP and BQP because the quantum algorithm solves the problem with $n^0=1$ queries while the best probabilistic classical algorithm requires $n$ queries. In contrast, Simon's Problem studied in Sec. 5.7 provides an exponential oracle separation from the best probabilistic (or deterministic) classical algorithm.

## 5.8 Grover Search Algorithm

Suppose that Bob's adversary, Alice, creates an unstructured database and Bob's task is to locate some particular data that he needs. For example, suppose the database comprises a large list of names and associated telephone numbers, but Alice has put the names in random rather than alphabetical order. To find the phone number of a particular person, there is no faster classical algorithm for Bob to use than to check each entry one at a time. If the total number of entries in the list is $N$, the average query complexity (number of times Bob has to look at the list) will be $N/2$ since the name he is looking for is equally likely to be at any position in the list. The worst-case scenario is that Bob will need $N$ queries. To subvert malicious intent on the part Alice, Bob might choose which entry to look at randomly using a probabilistic process. If Alice has no way of knowing the order in which Bob is going to search the list, she cannot force the worst-case scenario on Bob with certainty (though Bob may still be unlucky in his random choices and still end up needing $N$ queries). If Bob plans to use the database many times, he may find it useful to reorder the entries in the database using a sorting algorithm such as QUICKSORT which has an average-case cost of order $\sim N \log N$ and a worst-case cost of order $N^2$ queries.

Remarkably, Lov Grover (then at Bell Laboratories) invented a probabilistic quantum search algorithm that can find the answer with high probability in about $\sqrt{N}$ queries, something that is impossible classically!

To begin our analysis, consider the database illustrated in Table 5.3. In keeping with the previous algorithms we have studied, let us set this up as an oracle problem. Suppose that Bob is looking for the phone number of his friend Susan and let the binary encoding of her name be $\vec{Q}$. Let us define a function

$$f(\vec{x}) = \begin{cases} 1, \text{ if } \vec{Y}(\vec{x}) = \vec{Q}, \text{ else} \\ 0. \end{cases} \tag{5.116}$$

The function evaluates to 1 if the address $\vec{x}$ in the database corresponds to the entry for his friend's name. Otherwise it yields zero. Of course the function $f$ depends on the particular name $\vec{Q}$ that Bob is searching for, but we will leave this dependence implicit and keep only $\vec{x}$ as the argument of $f$.

Bob can query the function with different arguments $x$ until it returns 1 and then he only needs to look at position $x$ in the database to find the

| $\vec{x}$ | $\vec{Y}$ | $\vec{Z}$ |
|---|---|---|
| 000 | $\vec{Y}(000)$ | $\vec{Z}(000)$ |
| 001 | $\vec{Y}(001)$ | $\vec{Z}(001)$ |
| 010 | $\vec{Y}(010)$ | $\vec{Z}(010)$ |
| 011 | $\vec{Y}(011)$ | $\vec{Z}(011)$ |
| 100 | $\vec{Y}(100)$ | $\vec{Z}(100)$ |
| 101 | $\vec{Y}(101)$ | $\vec{Z}(101)$ |
| 110 | $\vec{Y}(110)$ | $\vec{Z}(110)$ |
| 111 | $\vec{Y}(111)$ | $\vec{Z}(111)$ |

Table 5.3: Unsorted database (of size 8) created by Alice. Here $\vec{x}$ is the binary number indicating the ordinal position in the database and $\vec{Y}(\vec{x})$ and $\vec{Z}(\vec{x})$ are a pair of binary numbers of fixed length $m$ representing the data. For example, $\vec{Y}(\vec{x})$ might be the ASCII encoding of a person's name, and $\vec{Z}(\vec{x})$ might be the binary encoding of their telephone number. The list of names is not sorted alphabetically but rather is in random order.

phone number he seeks. (Of course he could set up the function with extra bits in the output to automatically return the phone number if there is a match, but to simplify the notation we are leaving this aside.) As noted in Box 5.6, the important fact that Bob can use the database to create the oracle himself emphasizes the point that the oracle is *not* telling Bob the location of the data he is looking for–it is simply telling him whether or not the data is actually at the particular location that Bob queries the oracle about. That is, Bob is guessing the location in the database and the oracle is telling him whether or not his guess is correct.

**Box 5.6. Who can create the Grover Search Oracle?** It is interesting to recall that in the algorithms we have studied previously, it is the adversary Alice who creates the oracle function and Bob has access to it only as a 'black box.' Bob's task is to learn something about the hidden structure of the oracle function (i.e., a hidden bit string used to define the function). Here the situtation is different. Alice may have created the random ordering of the data in the database, but Bob can easily create the oracle function himself without knowing where the name he is looking for is in the database. He simply creates a function that takes in a value of $\vec{x}$ (and implicitly takes in the name Bob is searching for), goes to position with index $\vec{x}$ in the database and checks if the name there matches the name Bob is searching for. If it does match, the function returns 1, and if not, it returns 0.

To create a quantum algorithm we can, as usual, encode this function in a reversible unitary quantum oracle $\mathcal{O}_f$ that takes in the state

$$|\psi_{\text{in}}\rangle = |d\rangle \otimes |\vec{x}\,\rangle, \tag{5.117}$$

and outputs the state

$$|\psi_{\text{out}}\rangle = \mathcal{O}_f|\psi_{\text{in}}\rangle = |d \oplus f(\vec{x})\rangle \otimes |\vec{x}\,\rangle, \tag{5.118}$$

where $\vec{x}$ is the binary representation of the ordinal index in the database. For simplicity we will assume that $N = 2^n$, so that the range of indices can be encoded in bit strings $\vec{x}$ of length $n$. If necessary we can pad the database with additional entries (all of which return $f = 0$) to achieve this condition.

As in the Deutsch and Deutsch-Jozsa algorithm circuits (see Fig. 5.8) we will put the lower input wire $d$ into the superposition state $|-\rangle$ so that

$$|\psi_{\text{out}}\rangle = |-\rangle \otimes (-1)^{f(\vec{x})}|\vec{x}\,\rangle. \tag{5.119}$$

We see that the lower wire $d$ is a kind of quantum 'catalyst' that causes the phase kickback $(-1)^{f(\vec{x})}$ onto the input state, but is itself left unchanged in the process. Since the state of the lower wire is unchanged and is never entangled with the other wires, we will ignore it from now on and just consider the state of the upper $n$ wires. Let us suppose that the name $\vec{Q}$ that we are looking for is located at index $\vec{y}$ in the data base. That is, suppose that $\vec{Y}(\vec{y}) = \vec{Q}$. It is straight forward to see from Eq. (5.119) that the $(n+1) \times (n+1)$ oracle unitary $\mathcal{O}_f$ is equivalent (ignoring the catalyst wire) to the $n \times n$ unitary

$$U_f = I - 2|\vec{y}\,\rangle\langle\vec{y}\,| \tag{5.120}$$

where $I$ is the $n$-qubit identity operator. It is clear from this that

$$U_f|\vec{y}\,\rangle = -|\vec{y}\,\rangle, \text{ and} \tag{5.121}$$
$$U_f|\vec{x}\,\rangle = +|\vec{x}\,\rangle, \text{ if } \vec{x} \neq \vec{y}. \tag{5.122}$$

It is important here to again emphasize that even though the form of the oracle unitary as written in Eq. (5.120) depends explicitly on the 'answer' $\vec{y}$ that Bob seeks, Bob is still able to use the database to create the oracle himself without having to explicitly know the answer $\vec{y}$ in advance. Note that to do this, he has to use the extra qubit $d$ to create the reversible $(n+1)\times(n+1)$ unitary oracle matrix $\mathcal{O}_f$ based on the query success function $f$. It is only when we consider the special case that $d$ is initialized in the state $|-\rangle$ that is left unchanged at the end, that we can restrict our attention to the first $n$ qubits and write the effective $n \times n$ unitary matrix $U_f$. (See Box 5.6.)

Again, just as in the Deutsch-Jozsa algorithm, we will initialize the $n$ upper (input) wires in state

$$|\Phi_0\rangle = H^{\otimes n}|0\rangle^{\otimes n} = (H|0\rangle)^{\otimes n} = |+\rangle^{\otimes n}, \tag{5.123}$$

so that when we apply the oracle once we make one query, but it is in a superposition of all possible single queries. The result is

$$U_f|\Phi_0\rangle\big] = \big[I - 2|\vec{y}\,\rangle\langle\vec{y}\,|\big]|\Phi_0\rangle \tag{5.124}$$
$$= \frac{1}{\sqrt{2^n}}\sum_{\vec{x}}(-1)^{f(\vec{x})}|\vec{x}\,\rangle \tag{5.125}$$
$$= \frac{1}{\sqrt{2^n}}\sum_{\vec{x}\neq\vec{y}}|\vec{x}\,\rangle - \frac{1}{\sqrt{2^n}}|\vec{y}\,\rangle. \tag{5.126}$$

We see that the giant superposition state $|\Phi_0\rangle$ is unchanged except one entry (the one we want) has been 'marked' by the oracle because its sign flipped.

Now consider a function $g$ that is like $f$ except that it is for the specific case where the desired entry in the database has index $\vec{0}$. That is, $g(\vec{x}) = 1$ for $\vec{x} = \vec{0}$ but $g(\vec{x}) = 0$ for all $\vec{x} \neq \vec{0}$. Now encode this function in a unitary oracle (that we can create since know the function $g$) $U_g$. By the same argument as above

$$U_g = I - 2|\vec{0}\rangle\langle\vec{0}|. \tag{5.127}$$

Now use this to define the operator $G$ via

$$G = -H^{\otimes n} U_g H^{\otimes n} = 2|\Phi_0\rangle\langle\Phi_0| - I, \tag{5.128}$$

where the minus sign has been included purely for convenience.

The Grover algorithm consists of repeatedly applying the pair of operators

$$R \equiv GU_f \tag{5.129}$$

to the initial state $|\Phi_0\rangle$. Importantly, it turns out that since $R$ only involves the identity operator and the projectors $|\vec{y}\,\rangle\langle\vec{y}\,|$ and $|\vec{\Phi}_0\rangle\langle\Phi_0|$, $R$ is actually a rotation matrix in the space spanned by the starting state $|\Phi_0\rangle$ and the target state $|\vec{y}\,\rangle$. To see this, let us first find an orthonormal basis for this subspace. Since our target state is $|\vec{y}\,\rangle$, we will choose it as one basis state. $|\Phi_0\rangle$ is very nearly orthogonal to $|\vec{y}\,\rangle$

$$g \equiv \langle\vec{y}\,|\Phi_0\rangle = \frac{1}{\sqrt{2^n}}. \tag{5.130}$$

Thus the second basis vector is very close to $|\Phi_0\rangle$ (up to an unimportant arbitrary global phase that we are free to choose). Let us define the second basis vector to be

$$|\chi\rangle = \frac{1}{\sqrt{W}}\left[|\Phi_0\rangle - g|\vec{y}\,\rangle\right], \tag{5.131}$$

where $W = 1 - g^2$ is the normalization constant. It is straightforward to see that this choice guarantees that this second basis vector is indeed orthogonal to the first

$$\langle\vec{y}\,|\chi\rangle = 0. \tag{5.132}$$

Using the orthonormality of the basis, an arbitrary superposition of states in the span of these two basis vectors can be written

$$|\psi\rangle = \cos\frac{\theta}{2}|\chi\rangle + \sin\frac{\theta}{2}e^{i\varphi}|\vec{y}\,\rangle, \tag{5.133}$$

thereby defining a point with coordinates $\theta, \phi$ on an effective 'Bloch sphere' shown in Fig. 5.13 that has $|\chi\rangle$ at the north pole and $|\vec{y}\,\rangle$ at the south pole. As an example, the input state for the algorithm is from Eq. (5.131)

$$|\Phi_0\rangle = \sqrt{W}|\chi\rangle + g|\vec{y}\,\rangle, \tag{5.134}$$

and is described by angles $\theta_0, \varphi_0$ obeying

$$\sin\frac{\theta_0}{2} = g \tag{5.135}$$

$$\cos\frac{\theta_0}{2} = \sqrt{W} \tag{5.136}$$

$$\varphi_0 = 0. \tag{5.137}$$

Since (for large $n$) $g \ll 1$, the initial state is nearly orthogonal to the target state $|\vec{y}\,\rangle$, and thus is extremely close to the north pole. Further more it lies in the $XZ$ plane of the Bloch sphere as shown in Fig. 5.13. The initial state $|\Phi_0\rangle$ contains the target state, but with only a very small amplitude so that a measurement is exponentially unlikely to yield the desired answer. The goal of the Grover algorithm is to 'amplify' the quantum amplitude of the target state to near unity. If this can be achieved, then measurement of the output register will yield the database address $\vec{y}$ corresponding to the name of Bob's friend. With this information Bob then knows where to look to find his friends phone number.

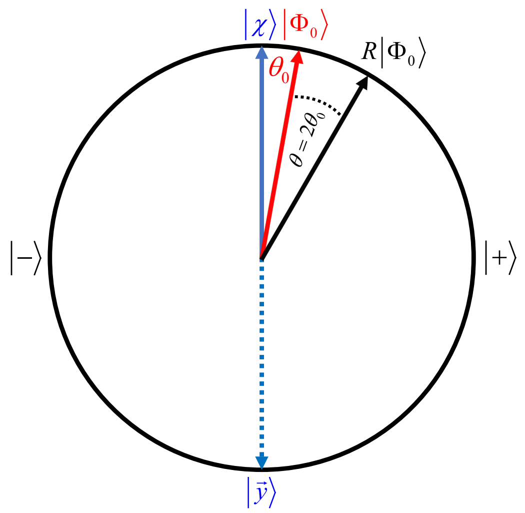

Figure 5.13: Effective Bloch sphere for the Grover search algorithm for the two-dimensional subspace spanned by the starting state $|\Phi_0\rangle$ and the target state $|\vec{y}\,\rangle$. The target state lies at the south pole and the starting state lies very close to the north pole, at exponentially small polar angle $\theta_0$. Application of the rotation $R$ increase the polar angle by $2\theta_0$.

**Exercise 5.4.** Show that the basis state orthogonal to $|\vec{y}\,\rangle$ in Eq. (5.131) is given by

$$|\chi\rangle = \frac{1}{\sqrt{2^n-1}}\sum_{\vec{x}\neq\vec{y}}|\vec{x}\,\rangle.$$

Consider the action of $U_f$ on the basis states

$$U_f|\chi\rangle = +|\chi\rangle, \tag{5.138}$$
$$U_f|\vec{y}\,\rangle = -|\vec{y}\,\rangle. \tag{5.139}$$

What is the effect of $G$ in this subspace? We see that

$$G|\chi\rangle = -|\chi\rangle + 2|\Phi_0\rangle\langle\Phi_0|\chi\rangle \qquad = (2W-1)|\chi\rangle + 2g\sqrt{W}|\vec{y}\,\rangle, \tag{5.140}$$
$$G|\vec{y}\,\rangle = 2|\Phi_0\rangle\langle\Phi_0|\vec{y}\,\rangle - |\vec{y}\,\rangle \qquad = 2g\sqrt{W}|\chi\rangle - (2W-1)|\vec{y}\,\rangle. \tag{5.141}$$

Combining all these results we have that the effect of $R = GU_f$ is

$$R|\chi\rangle = (2W-1)|\chi\rangle + 2g\sqrt{W}|\vec{y}\,\rangle, \tag{5.142}$$
$$R|\vec{y}\,\rangle = -2g\sqrt{W}|\chi\rangle + (2W-1)|\vec{y}\,\rangle. \tag{5.143}$$

Within this two-dimensional subspace we can use spinor representations for the basis states

$$\begin{pmatrix} 1 \\ 0 \end{pmatrix} = |\chi\rangle \tag{5.144}$$
$$\begin{pmatrix} 0 \\ 1 \end{pmatrix} = |\vec{y}\,\rangle, \tag{5.145}$$

and thus have the matrix representation of $R$ as a rotation around the $Y$ axis of the effective Bloch sphere by a small angle $+\theta$

$$R = \begin{pmatrix} \cos\frac{\theta}{2} & -\sin\frac{\theta}{2} \\ +\sin\frac{\theta}{2} & \cos\frac{\theta}{2} \end{pmatrix} = e^{-i\frac{\theta}{2}Y}, \tag{5.146}$$

with

$$\sin\frac{\theta}{2} = 2g\sqrt{W} = 2\sin\frac{\theta_0}{2}\cos\frac{\theta_0}{2} = \sin\theta_0, \tag{5.147}$$

where $\theta_0$ is the polar angle of the state $|\Phi_0\rangle$. Thus the rotation associated with $R$ is $\theta = 2\theta_0 = 4\arcsin g \approx 4g$, where for the last equality we have used the small angle approximation that is valid for large $n$. We see from Fig. 5.13 that since the rotation is about the $Y$ axis by a positive angle $\theta_0$, repeatedly applying $R$ rotates the Bloch vector closer and closer to the target state at the south pole. After $N$ applications of $R$ ($N$ oracle queries) the polar angle of the Bloch vector is

$$\theta_N = (2N+1)\theta_0. \tag{5.148}$$

The probability of the measured output state being $\vec{y}$ is maximized by choosing $\theta_N$ close to $\pi$ (so that the state of the system is close to $|\vec{y}\rangle$). This can be achieved by choosing $N$ to be an integer $N_\mathrm{q}$ close to

$$N_\mathrm{q} \approx \frac{\pi}{4\arcsin g} - \frac{1}{2} \approx \frac{\pi}{4}\sqrt{2^n}, \tag{5.149}$$

where again, the last approximate equality is valid for large $n$ (small $g$).

Thus we have the remarkable result that even though Bob does not know the target state, having access to the search oracle that can check if a guess is correct, allows Bob to rotate the initial state $|\Phi_0\rangle$ (almost perfectly) into the target state $|\vec{y}\rangle$! Again, the oracle does not tell Bob directly what the value of $\vec{y}$ is, only whether a guessed value $\vec{x}$ is correct or not.

The average case classical query complexity is $N_\mathrm{c} \sim 2^{n-1} \sim N_\mathrm{q}^2$ is quadratically worse than the quantum query complexity. This is often referred to as a quadratic quantum speed-up. Because the speed up is not exponential, the Grover algorithm must still call the oracle an exponential number of times

$$N_\mathrm{q} \sim 2^{\frac{n}{2}}. \tag{5.150}$$

In practice, with real-world quantum computers that suffer errors, it is difficult to avoid fatal errors for circuits that have exponential depth. Hence it is unlikely (at least in the near future without nearly perfect quantum error correction) that the Grover algorithm will have practical applications. Nevertheless, it is an interesting quantum programming primitive to keep in mind.

### 5.8.1 Geometric Interpretation of the Grover Operations

Recall the properties of the Pauli matrix $Z$ in terms of the standard single-qubit basis states

$$\begin{aligned} Z &\equiv |0\rangle\langle 0| - |1\rangle\langle 1| \\ &= \big[|0\rangle\langle 0| + |1\rangle\langle 1|\big] - 2|1\rangle\langle 1| \\ &= I - 2|1\rangle\langle 1|, \end{aligned} \tag{5.151}$$

where from the completeness relation, $I = \big[|0\rangle\langle 0| + |1\rangle\langle 1|\big]$ is the identity operator. This tells us that, while $U_f$ can act on the entire Hilbert space, when it acts on the two-dimensional manifold of states spanned by the Bloch sphere states we have defined, it acts like an effective Pauli $Z$

$$U_f = I - 2|\vec{y}\,\rangle\langle\vec{y}\,| = Z. \tag{5.152}$$

Similarly, the unitary $G$ also acts within the two-dimensional subspace like an effective Pauli matrix

$$G = 2|\Phi_0\rangle\langle\Phi_0| - I = +\vec{\lambda}\cdot\vec{\sigma}, \tag{5.153}$$

where $\vec{\lambda} = (\sin\theta_0, 0, \cos\theta_0)$ is the unit vector parallel to the spin vector associated with the location of $|\Phi_0\rangle$ on the Bloch sphere.

Now notice that up to an irrelevant global phase factor, $Z$ produces a rotation of the spin around the $Z$ axis by an angle of $\pi$. Thus $U_f$ does the same. Similarly, G is equivalent to a rotation around the $\vec{\lambda}$ axis. This leads us to a different geometric understanding of the Grover amplification process illustrated in Fig. 5.14.

### 5.8.2 Multiple Entry Search

We can readily generalize the Grover algorithm to the case where the database contains multiple targets for the search. For example, suppose that Bob is searching for anyone in the database with a certain family name. How does he modify his quantum search if there are $M$ entries in the database for which the function $f$ returns the value 1? Rather than a single state $|\vec{y}\,\rangle$, we can take seek a target state

$$|T_M\rangle = \frac{1}{\sqrt{M}}\sum_{\vec{x}} f(\vec{x})|\vec{x}\,\rangle. \tag{5.154}$$

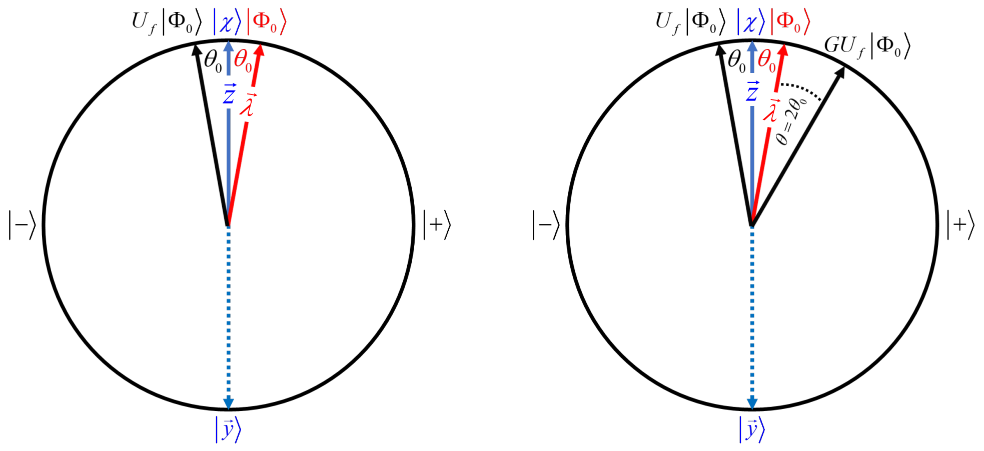

Figure 5.14: Grover amplification process. Left panel shows that $U_f$ rotates the Bloch vector $\vec{\lambda}$ of state $|\Phi_0\rangle$ by $\pi$ around the $Z$ axis. Right panel shows the next step in which $G$ rotates the state vector by $\pi$ around the $\vec{\lambda}$ axis. The combined effect increases the polar angle of the state vector from $\theta_0$ to $3\theta_0$, equivalent to a rotation around the $Y$ axis by an angle of $2\theta_0$.

Note that if $f(\vec{x}) = 1$ for $M$ distinct values of $\vec{x}$, then this state is properly normalized. If the algorithm were able to rotate the starting state $|\Phi_0\rangle$ into $|T_M\rangle$, then with high probability a single measurement will yield a random value $\vec{y}$ from among the $M$ distinct values. If Bob desires to obtain all $M$ of the solutions $\vec{y}_1, \ldots \vec{y}_M$ to the equation $f(\vec{x}) = 1$, he can do so probabilistically by running the search algorithm a number of times of order $M$ until it succeeds in finding all $M$ entries.

The operator $G$ and the initial state $|\Phi_0\rangle$ remain the same as in the previous case. The analog of Eq. (5.120) is

$$U_f = I - 2\sum_{j=1}^{M} |\vec{y}_j\rangle\langle\vec{y}_j|. \tag{5.155}$$

It is important to note that this operator is *not* the same as

$$V_f = I - 2|T_M\rangle\langle T_M|. \tag{5.156}$$

Nevertheless it is still true that

$$U_f|T_M\rangle = V_f|T_M\rangle = -|T_M\rangle, \tag{5.157}$$
$$U_f|\Phi_0\rangle = V_f|\Phi_0\rangle. \tag{5.158}$$

Hence within the subspace spanned by the initial state $|\Phi_0\rangle$ and the target state $|T_M\rangle$, the oracle $U_f$ is effectively equal to $V_f$.

The upshot is that the only change from the analysis for the case of $M = 1$ is the increase in the overlap between the initial and target states from $g$ to

$$\langle T_M|\Phi_0\rangle = \sqrt{\frac{M}{2^n}} = g\sqrt{M}. \tag{5.159}$$

Replacing $g$ in all the formulae by $g\sqrt{M}$ means that (for large $n$) the rotation angle $\theta$ increases by a factor of $\sqrt{M}$ and the rotation must be applied fewer times $N \approx \frac{\pi}{4}\sqrt{\frac{2^n}{M}}$ to optimize the success probability (of finding one of the correct entries) to be near unity. This again a quadratic quantum advantage over the classical query complexity $2^n/M$.

**Exercise 5.5.** Consider a Grover search problem where the database contains $M$ entries satisfying the search criterion (i.e. there are $M$ distinct indices for which the oracle query returns the value 1 instead of 0). Develop a probabilistic hybrid quantum/classical algorithm to find all $M$ distinct database entries. Hybrid in this case means that a classical computer keeps track of how many of the $M$ entries have been found so far and (may) decide to adjust the oracle or the algorithm accordingly.

### Oblivious Amplification

Notice that if we apply the Grover rotation operator $R$ in Eq. (5.129) too many or too few times, we will over or under rotate the state on the Bloch sphere and not end up close to the optimal position at the south pole. The correct number of times to apply $R$ is determined by the size of the database ($N_d = 2^n$) and the number of instances $M$ for which $f$ returns the value 1 (i.e., the number of 'correct answers'). We have to be given $N_d$ to be able to query the oracle. But what if the value of $M$ is unknown? Then we have a problem. Fortunately there is a clever modification of the Grover algorithm that solves this problem by means of oblivious amplitude amplification. That is, independent of the value of $M$, it automatically stops the process when it has brought the state near the south pole. [The method, introduced by Berry et al. in 2014, is reviewed in 'Fixed-point oblivious quantum amplitude-amplification algorithm,' Bao Yan et al., Scientific Reports volume 12, Article number: 14339 (2022). See also Appendix D of PRX QUANTUM 2, 040203 (2021)]

NEED TO PRESENT DETAILS OF METHOD.

### 5.8.3 Generic Quantum Speedup for Decision Problems

The Grover algorithm provides a very generic method to speed up decision problems where the answer is unknown or difficult to obtain, but is easy to check. As noted by Nielsen and Chuang, the factoring problem is one such example. Suppose we are given an exponentially large number $\Lambda$ that is the product of two (unknown) primes, $\Lambda = m_1 m_2$. Finding the prime factors is (believed to be) difficult. Conversely if you guess two factors, it is easy to multiply them to check if they are indeed factors. Rather than blindly guessing, one could develop a quantum oracle which takes as input two large numbers $M_1, M_2$ and returns 1 if their product equals $\Lambda$ and returns 0 otherwise. This would allow a Grover-accelerated search for the factors. However the speed up is actually not good relative to the best known classical algorithms which, while still superpolynomial,[2] are much better than random guessing. In addition, Shor's quantum algorithm (based on the quantum Fourier transform) solves the factoring problem vastly faster than the best known classical algorithms.

## 5.9 VQE and QAOA Algorithms

To be developed.

## 5.10 Phase Estimation Algorithm

To be developed.

## 5.11 Quantum Fourier Transform

To be developed.

[2]An example of superpolynomial scaling would be a function of the form $e^{(1/2)n^{(1/3)}}$, which grows more rapidly for large $n$ than any polynomial in $n$, but more slowly than exponentially in $n$, say as $e^{(1/16)n}$.

# Chapter 6

# Quantum Error Correction

IMPORTANT ADDITIONS TO BE MADE: (1) Explain how to create a circuit to measure stabilizers and (2) why learning both $Z_1$ and $Z_2$ would destroy the quantum information. Explain how quantum errors are continuous but measurement of error syndromes discretizes them. Explain how to do that with two CNOTs. Add discussion of repetition code for phase errors and then concatenate to get the Short code. Perhaps also present the toric code.

Now that we understand entanglement, we are in a position to tackle quantum error correction.

To overcome the deleterious effects of electrical noise, cosmic rays and other hazards, modern digital computers rely heavily on error correcting codes to store and correctly retrieve vast quantities of data. Classical error correction works by introducing extra bits which provide redundant encoding of the information. Error correction proceeds by measuring the bits and comparing them to the redundant information in the auxiliary bits. Another benefit of the representation of information as discrete bits (with 0 and 1 corresponding to a voltage for example) is that one can ignore small noise voltages. That is, $V = 0.99$ volts can be safely assumed to represent 1 and not 0.

All classical (and quantum) error correction codes are based on the assumption that the hardware is good enough that errors are rare. The goal is to make them even rarer. For classical bits there is only one kind of error, namely the bit flip which maps 0 to 1 or vice versa. We will assume for simplicity that there is probability $p \ll 1$ that a bit flip error occurs, and that error occurrences are uncorrelated among different bits. One of the simplest classical error correction codes to understand involves repetition and majority rule. Suppose we have a classical bit carrying the information we wish to protect from error and we have available two ancilla bits (also subject to errors). The procedure consists copying the state of the first bit into the two ancilla bits. Thus a 'logical' 1 is represented by three 'physical' bits in state 111, and a 'logical' 0 is represented by three 'physical' bits in state 000. Any other physical state is an error state outside of the logical state space.

Suppose now that one of the three physical bits suffers an error. For example, suppose that the state is supposed to be 000 but the first bit flips so the state becomes 001. Assuming that errors are rare and thus no more than 1 error has occurred, it is easy to identify that it is the first bit that flipped because it disagrees with the other two. That is, by examining the state of each bit it is a simple matter to identify the bit which has flipped and is not in agreement with the 'majority.' We then simply flip the minority bit so that it again agrees with the majority. This procedure succeeds if the number of errors is zero or one, but it fails if there is more than one error. For example the error state 001 could have arisen not from a single error in the code word 000 but rather two errors in the code word 111. There is no way to tell for sure which case actually occurred. However if errors are rare, it is far more likely that a single error occurred rather than two errors. Hence we should our error decoder should assume the more likely scenario. Similarly the state 000 could represent zero errors or three errors that 111 into 000. Again this is undetectable and causes a failure of the code. Let us now analyze this failure probability quantitatively.

Of course since we have replaced one imperfect bit with three imperfect bits, this means that the probability of an error occurring has increased considerably. However if the most frequent errors (a single bit flip on one of the three bits) are correctable, we may still come out ahead. For three bits the probability $P_n$ of $n$ errors is given by

$$\begin{aligned} P_0 &= (1-p)^3 &(6.1)\\ P_1 &= 3p(1-p)^2 &(6.2)\\ P_2 &= 3p^2(1-p) &(6.3)\\ P_3 &= p^3. &(6.4)\end{aligned}$$

Because our error correction code only fails for two or more physical bit errors the error probability for our logical qubit is

$$p_{\text{logical}} = P_2 + P_3 = 3p^2 - 2p^3, \tag{6.5}$$

The logical error rate is plotted against the physical probability in Fig. 6.1. We see that if the physical error probability per qubit obeys $p < 1/2$, then $p_{\text{logical}} < p$ and the error correction scheme reduces the error rate (instead of making it worse). Thus the 'break-even' point is $p^* = 1/2$. If the error probability is far below the break-even point, for example $p = 10^{-6}$, then $p_{\text{logical}} \sim 3p^2 \sim 3 \times 10^{-12}$. Thus the lower the raw error rate, the greater the improvement because the logical error rate scales quadratically $\sim 3p^2$. Note however that even at this low error rate, a petabyte ($8 \times 10^{15}$ bit) storage system would have on average 24,000 errors. Furthermore, one would have to buy three petabytes of storage since 2/3 of the disk would be taken up with ancilla bits!

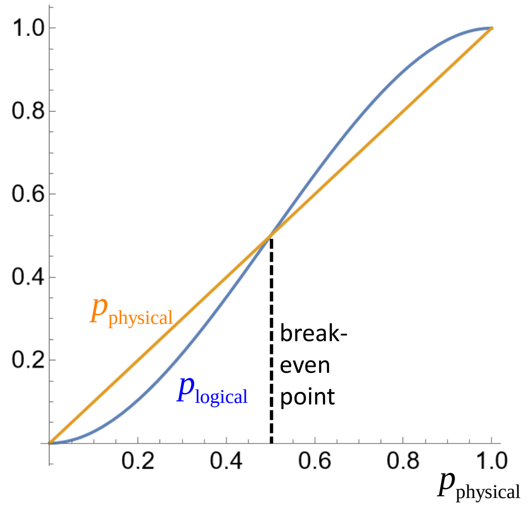

Figure 6.1: Plot of logical error probability vs. physical error probability per bit for the three bit repetition code. For small physical error probability, the logical error probability scales quadratically, $p_{\text{logical}} \sim 3p^2_{\text{physical}}$. The failure probability for the quantum repetition code is the same.

We are now ready to enter the remarkable and magic world of quantum error correction. Without quantum error correction, quantum computation would be impossible and there is a sense in which the fact that error correction is possible is even more amazing and counterintuitive than the fact of quantum computation itself. Naively, it would seem that quantum error correction is completely impossible. The no-cloning theorem (see Box 3.4) does not allow us to copy an unknown state of a qubit onto ancilla qubits. Furthermore, in order to determine if an error has occurred, we would have to make a measurement, and the back action (state collapse) from that measurement would itself produce random unrecoverable errors.

Part of the power of a quantum computer derives from its analog character–quantum states are described by continuous real (or complex) variables. This raises the specter that noise and small errors will destroy the extra power of the computer just as it does for classical analog computers. Remarkably, this is *not* the case! This is because the quantum computer also has characteristics that are digital. Recall that any measurement of the state of qubit always yields a binary result. Because of this, quantum errors are continuous, but *measured* quantum errors are discrete. Amazingly this makes it possible to perform quantum error correction and keep the calculation running even on an imperfect and noisy computer. In many ways, this discovery by Peter Shor in 1995 and Andrew Steane in 1996 is even more profound and unexpected than the discovery of efficient quantum algorithms that work on ideal computers.

It would seem obvious that quantum error correction is impossible because the act of measurement to check if there is an error would collapse the state, destroying any possible quantum superposition information. Remarkably however, one can encode the information in such a way that the presence of an error can be detected by measurement, and if the code is sufficiently sophisticated, the error can be corrected, just as in classical computation. Classically, the only error that exists is the bit flip. Quantum mechanically there are other types of errors (e.g. phase flip, energy decay, erasure channels, etc.). However codes have been developed (using a minimum of 5 qubits) which will correct all possible quantum errors. By concatenating these codes to higher levels of redundancy, even small imperfections in the error correction process itself can be corrected. Thus quantum superpositions can in principle be made to last essentially forever even in an imperfect noisy system. It is this remarkable insight that makes quantum computation possible.

As an entré to this rich field, we will consider a simplified example of one qubit in some state $\alpha|0\rangle + \beta|1\rangle$ plus two ancillary qubits in state $|0\rangle$ which we would like to use to protect the quantum information in the first qubit. As already noted, the simplest classical error correction code simply replicates the first bit twice and then uses majority voting to correct for (single) bit flip errors. This procedure fails in the quantum case because the no-cloning theorem (see Box 3.4) prevents replication of an unknown qubit state. Thus there does not exist a unitary transformation which takes

$$[\alpha|0\rangle + \beta|1\rangle] \otimes |00\rangle \longrightarrow [\alpha|0\rangle + \beta|1\rangle]^{\otimes 3}. \tag{6.6}$$

As was mentioned earlier, this is clear from the fact that the above transformation is not linear in the amplitudes $\alpha$ and $\beta$ and quantum mechanics is linear. One can however perform the repetition code transformation:

$$[\alpha|0\rangle + \beta|1\rangle] \otimes |00\rangle \longrightarrow [\alpha|000\rangle + \beta|111\rangle], \tag{6.7}$$

since this is in fact a unitary transformation. Just as in the classical case, these three physical qubits form a single logical qubit. The two logical basis states are

$$\begin{aligned} |0\rangle_{\text{log}} &= |000\rangle \\ |1\rangle_{\text{log}} &= |111\rangle. \end{aligned} \tag{6.8}$$

The analog of the single-qubit Pauli operators for this logical qubit are readily seen to be

$$\begin{aligned} X_{\text{log}} &= X_1X_2X_3 \\ Y_{\text{log}} &= iX_{\text{log}}Z_{\text{log}} \\ Z_{\text{log}} &= Z_1Z_2Z_3. \end{aligned} \tag{6.9}$$

We see that this encoding complicates things considerably because now to do even a simple single logical qubit rotation we have to perform some rather non-trivial three-qubit joint operations. It is not always easy to achieve an effective Hamiltonian that can produce such joint operations, but this is an essential price we must pay in order to carry out quantum error correction.

It turns out that this simple code cannot correct all possible quantum errors, but only a single type. For specificity, let us take the error operating on our system to be a single bit flip, either $X_1, X_2$, or $X_3$. These three together with the identity operator, $I$, constitute the set of operators that produce the four possible error states of the system we will be able to correctly deal with. Following the formalism developed by Daniel Gottesman, let us define two *stabilizer* operators

$$S_1 = Z_1Z_2 \tag{6.10}$$
$$S_2 = Z_2Z_3. \tag{6.11}$$

These have the nice property that they commute both with each other (i.e., $[S_1, S_2] = S_1S_2 - S_2S_1 = 0$) and with all three of the logical qubit operators listed in Eq. (6.9). This means that they can both be measured simultaneously and that the act of measurement does *not* destroy the quantum information stored in any superposition of the two logical qubit states. Furthermore they each commute or anticommute with the four error operators in such a way that we can uniquely identify what error (if any) has occurred. Each of the four possible error states (including no error) is an eigenstate of both stabilizers with the eigenvalues listed in the table below

| error | $S_1$ | $S_2$ |
|---|---|---|
| $I$ | $+1$ | $+1$ |
| $X_1$ | $-1$ | $+1$ |
| $X_2$ | $-1$ | $-1$ |
| $X_3$ | $+1$ | $-1$ |

Thus measurement of the two stabilizers yields two bits of classical information (called the 'error syndrome') which uniquely identify which of the four possible error states the system is in and allows the experimenter to correct the situation by applying the appropriate error operator, $I, X_1, X_2, X_3$ to the system to cancel the original error.

We now have our first taste of fantastic power of quantum error correction. We have however glossed over some important details by assuming that either an error has occurred or it hasn't (that is, we have been assuming we are in a definite error state). At the next level of sophistication we have to recognize that we need to be able to handle the possibility of a quantum superposition of an error and no error. After all, in a system described by smoothly evolving superposition amplitudes, errors can develop continuously. Suppose for example that the correct state of the three physical qubits is

$$|\Psi_0\rangle = \alpha|000\rangle + \beta|111\rangle, \tag{6.12}$$

and that there is some perturbation to the Hamiltonian such that after some time there is a small amplitude $\epsilon$ that error $X_2$ has occurred. Then the state of the system is

$$|\Psi\rangle = [\sqrt{1 - |\epsilon|^2} I + \epsilon X_2]|\Psi_0\rangle. \tag{6.13}$$

(The reader may find it instructive to verify that the normalization is correct.)

What happens if we apply our error correction scheme to this state? The measurement of each stabilizer will always yield a binary result, thus illustrating the dual digital/analog nature of quantum information processing. With probability $P_0 = 1 - |\epsilon|^2$, the measurement result will be $S_1 = S_2 = +1$. In this case the state collapses back to the original ideal one and the error is removed! Indeed, the experimenter has no idea whether $\epsilon$ had ever even developed a non-zero value. All she knows is that if there was an error, it is now gone. This is the essence of the Zeno effect in quantum mechanics that repeated observation can stop dynamical evolution. (It is also, once again, a clear illustration of the maxim that in quantum mechanics 'You get what you see.') Rarely however (with probability $P_1 = |\epsilon|^2$) the measurement result will be $S_1 = S_2 = -1$ heralding the presence of an $X_2$ error. The correction protocol then proceeds as originally described above. Thus error correction still works for superpositions of no error and one error. A simple extension of this argument shows that it works for an arbitrary superposition of all four error states.

## 6.1 An advanced topic for the experts

There remains however one more level of subtlety we have been ignoring. The above discussion assumed a classical noise source modulating the Hamiltonian parameters. However in reality, a typical source of error is that one of the physical qubits becomes entangled with its environment. We generally have no access to the bath degrees of freedom and so for all intents and purposes, we can trace out the bath and work with the reduced density matrix of the logical qubit. Clearly this is generically not a pure state. How can we possibly go from an impure state (containing the entropy of entanglement with the bath) to the desired pure (zero entropy) state? Ordinary unitary operations on the logical qubit preserve the entropy so clearly will not work. Fortunately our error correction protocol involves applying one of four possible unitary operations *conditioned on the outcome of the measurement of the stabilizers*. The wave function collapse associated with the measurement gives us just the non-unitarity we need and the error correction protocol works even in this case. Effectively we have a Maxwell demon which uses Shannon information entropy (from the measurement results) to remove an equivalent amount of von Neumann entropy from the logical qubit!

To see that the protocol still works, we generalize Eq. (6.13) to include the bath

$$|\Psi\rangle = [\sqrt{1-|\epsilon|^2}|\Psi_0, \mathrm{Bath}_0\rangle + \epsilon X_2]|\Psi_0, \mathrm{Bath}_2\rangle. \tag{6.14}$$

For example, the error could be caused by the second qubit having a coupling to a bath operator $\mathcal{O}_2$ of the form

$$V_2 = g\, X_2 \mathcal{O}_2, \tag{6.15}$$

acting for a short time $\epsilon\hbar/g$ so that

$$|\mathrm{Bath}_2\rangle \approx \mathcal{O}_2|\mathrm{Bath}_0\rangle. \tag{6.16}$$

Notice that once the stabilizers have been measured, then either the experimenter obtained the result $S_1 = S_2 = +1$ and the state of the system plus bath collapses to

$$|\Psi\rangle = |\Psi_0, \mathrm{Bath}_0\rangle, \tag{6.17}$$

or the experimenter obtained the result $S_1 = S_2 = -1$ and the state collapses to

$$|\Psi\rangle = X_2|\Psi_0, \mathrm{Bath}_2\rangle. \tag{6.18}$$

Both results yield a product state in which the logical qubit is unentangled with the bath. Hence the algorithm can simply proceed as before and will work.

Finally, there is one more twist in this plot. We have so far described a measurement-based protocol for removing the entropy associated with errors. There exists another route to the same goal in which purely unitary multi-qubit operations are used to move the entropy from the logical qubit to some ancillae, and then the ancillae are reset to the ground state to remove the entropy. The reset operation could consist, for example, of putting the ancillae in contact with a cold bath and allowing the qubits to spontaneously and irreversibly decay into the bath. Because the ancillae are in a mixed state with some probability to be in the excited state and some to be in the ground state, the bath ends up in a mixed state containing (or not containing) photons resulting from the decay. Thus the entropy ends up in the bath. It is important for this process to work that the bath be cold so that the qubits always relax to the ground state and are never driven to the excited state.

We could if we wished, measure the state of the bath and determine which error (if any) occurred, but in this protocol, no actions conditioned on the outcome of such a measurement are required.

Quantum error correction is extremely challenging to carry out in practice. In fact the first error correction protocol to actually reach the break even point (where carrying out the protocol extends rather than shortens the lifetime of the quantum information) was achieved by the Yale group in 2016. This was done not using two-level systems as qubits but rather by storing the quantum information in the states of a harmonic oscillator (a superconducting microwave resonator containing superpositions of $0, 1, 2, \ldots$ photons).

# Chapter 7

# Yet To Do

1. Dave Bacon lecture notes on reversible classical gates has a nice discussion of Charlie Bennett's "uncomputation" trick to erasure scratch registers. Add a discussion of this.

2. Add more discussion of joint measurements with examples for students to make sure they understand that $Z \otimes X$ is not measured by measuring $Z \otimes I$ and $I \otimes X$ since this give too much information. However the expectation value of this operator can be measured from averaging the product of the individual measurement results. Another example is measuring $XY = iZ$ which is NOT the product of the individual measurements since they are incompatible.

3. Define mutual information, useful for QEC.

4. Important: For classical information theory present argument from Raisbeck's book to derive the Shannon formula when the probabilities are not all equal. Brings in concept that Shannon is the average information in a message, not necessarily the information in a particular message.

5. Inconsistent notation for states $|\pm Z\rangle, |\pm X\rangle, |0\rangle, |1\rangle, |+\rangle, |-\rangle$. See for example section where Hadamard gate is introduced. Need to make notation uniform or at least explain the equivalences among these states.

6. Discuss projective measurements in terms of projectors. Specifically discuss projective measurements for only a subset of the qubits (so the projector contains some identity operators). Let the reader know

that this is important for Simons's algorithm and for quantum error correction.

7. Perhaps introduce QM as a theory of probability by showing that the probability of some event classically is the sum of the probabilities of independent routes to the event. For example P('A or B') is P(A)[1-P(B)]+[1-P(A)]P(B)+P(A)P(B)=P(A)+P(B)-P(A)P(B) whereas in quantum you have to add amplitudes before you square

8. Add to the appendix on statistics discussion of conditional probabilities, factoring of probability distributions, meaning of statistical independence, etc.

9. Insert 2019 homework as exercises in text

10. Did I insert proof that length preservation implies general inner product conservation which implies unitary? Proof could be improved.

11. Appendix on linear algebra, outer products, direct sums, eigenvalues, degeneracies, determinants, traces, hermitian matrices producing bases, etc.

12. 3 CNOTS make a SWAP but apply to a general state or to the X basis

13. quantum teleportation

14. Shor code as next step beyond 3 qubit repetition code

15. matrix mechanics for harmonic oscillator?

16. Make sure this statement: "Peculiar quantum interference effects permit the Toffoli gate to be synthesized from two-qubit gates, something that is not possible in (reversible) classical computation. More on this later!" in Chapter 1 gets followed up. Show the quantum synthesis of Toffoli from CNOTs.

17. In Chap 2 when I introduce entanglement I mention the special type of entanglement called 'magic' and say we will look into it further later. Make sure this happens.

18. SECTION 3.0: Of course if the system is in a superposition of two orthogonal states, the measurement result can be random. Conversely, if a system is in one of two states that are not orthogonal, it is not possible to reliably determine by measurement which state the system is in. We shall explore this more deeply when we discuss measurements. MAKE SURE TO DO THIS.

19. Need to make sure that we derive (perhaps in appendix B) that eigenvectors of any Hermitian operator form a complete basis. Then give an exercise where students derive that the variance of measurement results of an operator is $\langle\psi|\hat{Q}^2|\psi\rangle - \langle\psi|\hat{Q}|\psi\rangle^2$.

20. Add derivation of completeness relation and eigenvectors of a Hermitian operator are complete to App. B. Box 3.2 states that this will be in App. B.

21. Add discussion of classical Euler angles to Sec. 3.4 and show that in quantum a final rotation along the qubit polarization axis just introduces a global phase on the state. [See Lecture 08 spring 2023.]

22. Consider adding a chapter on quantum channels in connection with error connection.

23. in Chapter 3 or maybe appendix B, do a better job of defining the differences among tensor, Kronecker and outer products.

24. Redo Box and Figure on physical implementation of CNOT to use $|0\rangle$ and $|1\rangle$ instead of $|\uparrow\rangle$ and $|\downarrow\rangle$. Also change sign of Hamiltonian.

25. Include section on entanglement distillation (referred to in Chap. 4). Needs to be after errors are discussed.

26. Tighten up mixture of notation among $\sigma^x$, $X$, $|+Z\rangle, |+\rangle$, etc.

27. create a Box to explain that Toffoli is universal for classical computation and Toffoli + Hadamard is universal for quantum (Dorit paper). Classically the Toffoli cannot be synthesized from CNOTs and NOTs but quantum mechanically it CAN be synthesized from CNOTs and single qubit (non-Clifford) rotations.

28. Define the Clifford hierarchy somewhere. Perhaps when discussing stabilizer codes in QEC.

29. ~~IMPORTANT: Show that the space of binary strings is a vector space over the field $\{0, 1\}$. The only allowed scalars are 0 and 1, and vectors are added bitwise mod 2. Refer to this when we do the Deutsch and the Deutsch-Jozsa algorithms.~~

30. Review JGEH comments on Box 2.2 on configuration space vs state space vs phase space. See his email 2023.02.26 3:01pm.

# Appendix A

# Quick Review of Probability and Statistics

## A.1 Randomness

Randomness plays an essential role in the theory of information–both classical and quantum. It is therefore useful for us to review basic concepts from probability and statistics.

What is randomness anyway? In the classical world randomness is related to ignorance. We lack knowledge of all the conditions and parameters needed to make accurate predictions. For example, flipping a coin and seeing if it lands face up or face down is considered random. But it isn't really random. If Bob watches Alice flip a coin and were able measure exactly how rapidly she made it spin and measure its initial upward velocity, he could predict (using Newton's laws of classical dynamics) how long it will be in the air and whether it will land face up or face down. In more complicated dynamical systems with several interacting degrees of freedom, the motion can be chaotic. Tiny changes in initial conditions (positions and velocities) can lead to large changes in the subsequent trajectory. Thus even though classical mechanics is completely deterministic, motion on long time scales can appear to be random.

Many computer programs rely on so-called random number generators. They are not actually random but rather chaotic iterative maps–they take an initial seed number and compute some complicated function of that seed to produce a new number. That number is then used as the input to the next round of iteration. The results may look random and may even pass many statistical tests for randomness, but if an observer knows the program that was used and the starting seed, he or she can predict the entire sequence of numbers perfectly.

In quantum mechanics randomness is an ineluctable feature. It is not due to ignorance of initial conditions but rather is an essential part of the theory. Alice can prepare $N$ truly identical copies of a quantum state and Bob can make a measurement of some physical property on each copy of that state and obtain truly random (not pseudo-random) results. The results are not random because of some 'hidden variable' whose value Alice forgot to fix or Bob failed to measure. They are truly random–it is impossible to predict the outcome of the measurement before it is performed.

## A.2 Probabilities

The probability $p_j$ of an event $j$ is a non-negative number obeying $0 \leq p_j \leq 1$. The probabilities of all possible events (in a universe of $M$ possible events) obeys the sum rule

$$\sum_{j=1}^{M} p_j = 1. \tag{A.1}$$

This simply means that one of the possible events (from the complete set of all possible mutually exclusive events) definitely happened.

As an example, suppose we have an $M$-sided die (die is the singular form of dice). On each face of the die is a number. Let $x_j$ denote the number of the $j$th face of the die, and $p_j$ be the probability that the $j$th face lands face up so that the number $x_j$ is showing when the die is randomly tossed onto a table. We can define a so-called 'random variable' $X$ to be the number that comes up when we toss the die. $X$ randomly takes on one of the allowed values $x_j; j = 1, \ldots, M$. We can now ask simple questions like, what is the mean (i.e. average) value of $X$? This is also known as the expectation value of $X$ and is often denoted by an overbar or by double brackets

$$\bar{X} = \langle\langle X \rangle\rangle = \sum_{j=1}^{M} p_j x_j. \tag{A.2}$$

This sum over all possible results weighted by their frequency of occurrence gives the value one would obtain by averaging the results of a huge number of trials of the experiment (of rolling the die).

As an example, suppose we have a standard cube-shaped die with the six faces numbered 1 through 6, that is $x_j = j$. We will take the 'measurement result' to be the number on the top face of the die after it stops rolling. If the die is fair, the probability of result $x_j$ is $p_j = \frac{1}{6}$, and thus the mean value will be

$$\bar{X} = \sum_{j=1}^{6} p_j x_j = \sum_{j=1}^{6} \left(\frac{1}{6}\right) j = 3.5. \tag{A.3}$$

**Exercise A.1.** Consider a pair of standard six-sided die with faces numbered consecutively from 1 to 6. Assuming the dice are fair

a) What are the unique possible values for the sum of the two numbers showing on the top faces of the dice?

b) What is the probability that each of these unique possible values occurs?

For later purposes, it will also be useful to consider what happens when we have two independent rolls of the die. Let the random variable $X_1$ be the number that comes up on the first toss and let $X_2$ be the number that comes up on the second toss. What is the joint probability distribution for the two results? That is, what is the probability $\mathcal{P}(x_j, x_k)$ that $X_1 = x_j$ and $X_2 = x_k$? Because each result is drawn independently from the same probability distribution we simply have that the probability for a given result of two tosses is just the product of the probabilities of the individual results

$$\mathcal{P}(x_j, x_k) = p_j p_k. \tag{A.4}$$

This is simply the statement that the joint probability distribution *factorizes* into two separate distributions for the individual events (assuming the events are independent of each other). From this it follows that

$$\begin{aligned}
\langle\langle X_1 X_2\rangle\rangle &= \sum_{j=1}^{M}\sum_{k=1}^{M}\mathcal{P}(x_j,x_k)x_j x_k \\
&= \sum_{j=1}^{M}\sum_{k=1}^{M} p_j p_k x_j x_k \\
&= \left\{\sum_{j=1}^{M} p_j x_j\right\}\left\{\sum_{k=1}^{M} p_k x_k\right\} \\
&= \langle\langle X_1\rangle\rangle\langle\langle X_2\rangle\rangle = \bar{X}^2. \qquad (A.5)
\end{aligned}$$

We conclude that for independent (i.e. uncorrelated) random variables, the mean of the product is equal to the product of the means.

The random variables $X_1, X_2$ will not be independent if they correspond to the result from the *same* throw of the die. Then of course $X_2 = X_1 = X$ and we have a different result

$$\langle\langle (X)^2\rangle\rangle = \sum_{j=1}^{M} p_j\,(x_j)^2\,. \qquad (A.6)$$

This is simply the mean of the square of the number that comes up when we toss the die. Does the mean of the square bear any relation to the square of the mean, $\bar{X}^2$? To find out, let us consider the so-called *variance* of the distribution, the mean of the square of the deviation of the random variable from its own mean

$$\begin{aligned}
\sigma^2 \equiv \langle\langle \left(X-\bar{X}\right)^2\rangle\rangle &= \sum_{j=1}^{M} p_j\left(x_j-\bar{X}\right)^2 \\
&= \sum_{j=1}^{M} p_j\left\{(x_j)^2 - 2\bar{X}x_j + \bar{X}^2\right\} \\
&= \langle\langle X^2\rangle\rangle - \langle\langle X\rangle\rangle^2. \qquad (A.7)
\end{aligned}$$

In deriving this result we have used the fact that

$$\sum_{j=1}^{M} p_j\bar{X}^2 = \bar{X}^2\sum_{j=1}^{M} p_j = \bar{X}^2, \qquad (A.8)$$

and

$$\sum_{j=1}^{M} p_j 2\bar{X} x_j = 2\bar{X} \sum_{j=1}^{M} p_j x_j = 2\bar{X}^2. \tag{A.9}$$

Clearly the variance is non-negative $\sigma^2 \geq 0$ because $(X - \bar{X})^2$ can never be negative. Hence we conclude

$$\langle\langle X^2 \rangle\rangle \geq \langle\langle X \rangle\rangle^2. \tag{A.10}$$

The variance is a measure of the *width* of the probability distribution. Another related quantity is the standard deviation or 'root mean square deviation' of the random variable from its mean

$$\sigma \equiv \langle\langle \left(X - \bar{X}\right)^2 \rangle\rangle^{\frac{1}{2}}. \tag{A.11}$$

To understand better what we mean by the width of the distribution consider the following examples. If a six-sided die has the number 3 painted on every face the probability distribution has zero variance. The only number that ever comes up is 3 so the mean of the distribution is 3 and no result ever deviates from the mean. Similarly, if the die has the standard consecutive numbering of the faces from 1 to 6, then a wide variety of outcomes is possible. If the die is fair then all outcomes are equally likely and the probability distribution is wide as shown in the left panel of Fig. A.1. Suppose however that the die is unfair and yields the number 3 with probability $p_3 = 0.8$, and the other numbers with probability $p_1 = p_2 = p_4 = p_5 = p_6 = 0.04$. This distribution is plotted in the right panel of Fig. A.1. The distribution has wide support (i.e. is non-zero over the full range from 1 to 6) but is still sharply peaked at 3. Hence the variance is small but not zero.

**Exercise A.2.** A standard six-sided die has faces numbered consecutively from 1 to 6. Find the variance and standard deviation of the probability distribution associated with random throws of the die

a) Assuming the die is fair (as in the left panel of Fig. A.1).

b) Assuming the die is biased with the probabilities given in the right panel of Fig. A.1.

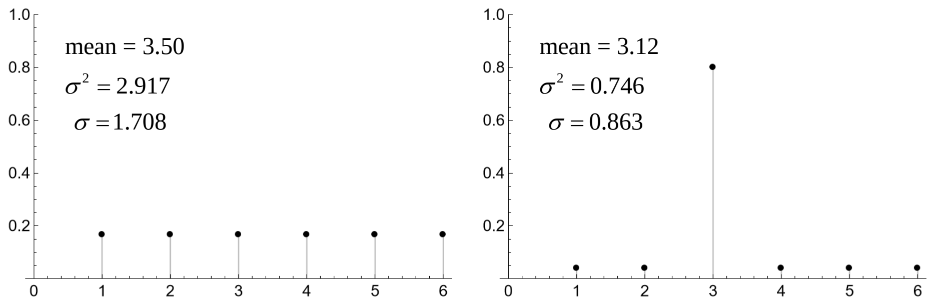

Figure A.1: Left panel: Graph of the probability distribution for the outcome of the throw of a fair die ($p_j = 1/6; j = 1, \ldots, 6$). The variance is large. Right panel: Graph of the probability distribution of an unfair (highly biased) die having $p_3 = 0.8$, and $p_1 = p_2 = p_4 = p_5 = p_6 = 0.04$. The variance is smaller.

## A.3 Statistical Estimators

When performing experiments one may wish to attempt to measure the average value of some statistical quantity. Since we can execute only a finite number of runs of the experiment, we cannot be guaranteed to obtain the exact value of the mean, only an estimate.

For $N$ trials we can form a so-called 'estimator' $\tilde{X}$ for $\bar{X}$, the mean value of a random variable via

$$\tilde{X}(N) = \frac{1}{N}\sum_{k=1}^{N} X_k, \tag{A.12}$$

where $X_k$ is the number that came up in the $k$th run of the experiment (e.g. throw of the die). We use the tilde over the $X$ to indicate that this is an estimator for the mean, not the true mean.

For finite $N$, this estimator is not likely to be exact, but for large $N$ we expect it to become better and better. Is there a way to determine how accurate this estimator is likely to be? Indeed there is. Let us write

$$\tilde{X}(N) = \bar{X} + \delta_N, \tag{A.13}$$

where the error $\delta_N$ is given by

$$\delta_N = \frac{1}{N}\sum_{k=1}^{N}\left[X_k - \bar{X}\right] \tag{A.14}$$

If we were to ensemble average this estimator over a huge (formally infinite) number of experiments consisting of $N$ throws we would obtain

$$\langle\langle \delta_N \rangle\rangle = \frac{1}{N}\sum_{k=1}^{N}\left[\langle\langle X_k\rangle\rangle - \bar{X}\right] = 0. \tag{A.15}$$

Thus the average error vanishes. That is, our estimator is *unbiased* as expected.

We can get a sense of the typical size of the error by considering its variance:

$$\begin{aligned}
\sigma_N^2 &\equiv \langle\langle (\delta_N)^2 \rangle\rangle \\
\sigma_N^2 &= \frac{1}{N^2}\sum_{j=1}^{N}\sum_{k=1}^{N}\langle\langle \left[X_j - \bar{X}\right]\left[X_k - \bar{X}\right]\rangle\rangle \\
&= \frac{1}{N^2}\sum_{j=1}^{N}\sum_{k=1}^{N}\left\{\langle\langle X_j X_k\rangle\rangle - \bar{X}^2\right\} \\
&= \frac{1}{N^2}\sum_{j\neq k}^{N}\left\{\langle\langle X_j X_k\rangle\rangle - \bar{X}^2\right\} + \frac{1}{N^2}\sum_{j=k}^{N}\left\{\langle\langle X_j X_k\rangle\rangle - \bar{X}^2\right\} \\
&= \frac{1}{N^2}\sum_{j=1}^{N}\left\{\langle\langle (X_j)^2\rangle\rangle - \bar{X}^2\right\} \\
&= \frac{1}{N}\sigma_1^2, \qquad \text{(A.16)}
\end{aligned}$$

where $\sigma_1 = \sigma$, the standard error for a single throw of the die defined in Eq. (A.11). Thus our estimator has a random error whose variance decreases inversely with $N$. The standard error thus is

$$\sigma_N = \frac{1}{\sqrt{N}}\sigma_1. \tag{A.17}$$

This is a simple estimate of the size of the error that our estimator of the mean is likely to have.

**Box A.1. Sample variance vs. true variance** Care must be exercised when estimating the variance $\sigma_1^2$ of an unknown probability distribution from a finite sample size drawn from the distribution. If (somehow) we know the true mean of the distribution, then we can simply use as our estimator of the variance

$$\tilde{\sigma}^2 = \frac{1}{N}\sum_{j=1}^{N}(X_j - \bar{X})^2. \tag{A.18}$$

It is straightforward to show that this is an unbiased estimator since

$$\langle\langle\tilde{\sigma}^2\rangle\rangle = \sigma_1^2. \tag{A.19}$$

The situation is not so simple when we do not know the mean of the unknown distribution and are forced to estimate it using Eq. (A.12). While this estimator of the mean is unbiased it still will have some small error and that error is positively correlated with the values of $X_j$ in our sample. This means that if we use as our estimator of the variance

$$\tilde{\sigma}^2 = \frac{1}{N}\sum_{j=1}^{N}(X_j - \tilde{X}(N))^2, \tag{A.20}$$

it will be biased because it is too small. As an extreme example, consider the case $N = 1$. Our estimate of the mean is $\tilde{X}(N) = X_1$. If we substitute this for the mean in Eq. (A.20) we always obtain $\tilde{\sigma}^2 = (X_1 - X_1)^2 = 0$. A straightforward calculation shows that for general $N$

$$\langle\langle\tilde{\sigma}^2\rangle\rangle = \frac{N-1}{N}\sigma_1^2, \tag{A.21}$$

which is consistent with our result that it vanishes for $N = 1$. Therefore if we want to have an unbiased estimate of the variance we should use

$$\tilde{\sigma}_S^2 = \frac{N}{N-1}\tilde{\sigma}^2 = \frac{1}{N-1}\sum_{j=1}^{N}(X_j - \tilde{X}(N))^2.. \tag{A.22}$$

We will refer to this as the unbiased sample variance.

Let us return now to the question of estimating the mean of a distribution from $N$ samples. We have an unbiased estimator and have already computed the variance of our estimator in Eq. (A.17). The question arises as to whether or not we could say something about the probability distribution of the error. For large $N$, the error $\delta_N$ in Eq. (A.14) is the sum of a large number of small random terms. By the *central limit theorem*, the error will be, to a good approximation, Gaussian distributed. That is, the probability distribution for the error is well approximated by a *continuous* distribution having *probability density*

$$P(\delta_N) = \frac{1}{\sqrt{2\pi\sigma_N^2}} e^{-\frac{1}{2\sigma_N^2}\delta_N^2}. \tag{A.23}$$

The interpretation of probability density for a continuous variable $x$ is the following. $P(x)dx$ is the probability that the value of the random variable $X$ lies between $x$ and $x + dx$. The normalization condition on probability becomes an integral (see the discussion on Gaussian integrals in Box A.2)

$$\int_{-\infty}^{+\infty} dx\, P(x) = 1. \tag{A.24}$$

Summations over random variables such as in Eq. (A.14) can be interpreted as random walks. Suppose that $\Delta$ is a random variable with equal probability of being $\pm\epsilon$, where $\epsilon$ is a fixed step length. Then a random walk of $N$ steps ends up a position

$$x = \sum_{j=1}^{N} \Delta_j, \tag{A.25}$$

where $\Delta_j$ is the value of the random variable $\Delta$ for the $j$th step.

In Fig. A.2 we see plots of the probability distribution for $x$ for different values of $N$, together with the Gaussian approximation to it. We see that the Gaussian approximation is quite good even for modest values of $N$. This is the essence of the central limit theorem. The sum of a large number of random variables (with bounded variance) is well-approximated by a Gaussian distribution.

Exercise A.3 provides an opportunity to prove the central limit theorem for the particular case of a random walk. To see how this works, let us derive the *exact* probability distribution for a random walk of $N$ steps, each of size $\epsilon = 1/2$. We take $N$ even so that the final position $m$ of the walker is always an integer and lies in the interval $m \in [-N/2, +N/2]$. Let $p_\pm$ be the probability of stepping to the right or left respectively. Let the number of steps to the right be $R$ and the number to the left be $L$. We have $N = R + L$ and the final position is given by $m = (R - L)\epsilon$. The probability of any given sequence of steps is

$$P = p_+^R p_-^L. \tag{A.26}$$

The number of different walks with $R$ steps to the right and $L$ steps to the left can be determined from combinatorics. Think of a string of $N$ symbols, $\ell$ and $r$ denoting the direction of each step in the random walk. There are altogether $N!$ permutations of the order of these symbols. However $L!$ permutations merely swap $\ell$'s with other $\ell$'s and so should not be counted as distinct walks. Similarly there are $R!$ permutations of the $r$'s that should not be counted. The number of distinct walks $M(R, L)$ is therefore

$$M(R, L) \quad = \quad \frac{N!}{R!L!}. \tag{A.27}$$

The probability of ending up at $m = (R - L)\epsilon$ after $N = R + L$ (even) steps is therefore

$$P(N, m) = p_+^R p_-^L \frac{N!}{R!L!} = p_+^R p_-^{(N-R)} \left( \begin{array}{c} N \\ R \end{array} \right) \tag{A.28}$$

where the last expression is the binomial coefficient

$$\left( \begin{array}{c} N \\ R \end{array} \right) = \frac{N!}{R!(N - R)!}. \tag{A.29}$$

For this reason, this probability distribution is known as the binomial distribution. Notice that this expression is correctly normalized because

$$\sum_{m=-N}^{+N} P(N, m) = \sum_{R=0}^{N} p_+^R p_-^{(N-R)} \frac{N!}{R!(N - R)!} = (p_+ + p_-)^N = (1)^N = 1, \tag{A.30}$$

where we have used the binomial theorem to evaluate the sum.

If the final position is $m = (R - L)\epsilon$, we have (for $\epsilon = 1/2$) $R = (N + m)/2$ and $L = (N - m)/2$ so Eq. (A.28) can also be written

$$P(N, m) = p_+^{(N+m)/2} p_-^{N-m)/2} \frac{N!}{[(N + m)/2]!\,[(N - m)/2]!} \tag{A.31}$$

**Exercise A.3.** A random walker moving in one dimension takes steps of length $\epsilon = 1/2$ to the right ($x \to x + \epsilon$) with probability $p$ and to the left ($x \to x - \epsilon$) with probability $q = 1 - p$. The walker starts at $x = 0$ and the position after $N$ steps is simply the algebraic sum of all the steps. The final position lies in the interval $[-N\epsilon, +N\epsilon]$. If $\epsilon = 1/2$ and $N$ is even, only integer positions are possible.

a) *Derive* exact expressions for the mean and variance of the position $m$ after $N$ steps by two different methods. Hint: You can either derive the appropriate properties of the binomial distribution, or you can use the fact that the individual walk steps have a mean and variance and are statistically independent. For the former it is useful to notice that $p_+ \frac{\partial}{\partial p_+}(p_+)^R = R(p_+)^R$.

b) Discuss and explain what happens to the variance as $p$ approaches either 0 or 1.

c) Take the logarithm of $P(N, m)$ in Eq. (A.31) and use Stirling's (asymptotic*) expansion

$$\ln Z! \sim Z \ln Z - Z + \frac{1}{2}\ln(2\pi Z) \tag{A.32}$$

for the factorials in the binomial coefficients to show that for large $N$ the binomial distribution with $q = p = 1/2$ approaches the Gaussian distribution

$$P_N(x) = \frac{1}{\sqrt{2\pi\sigma^2}} e^{-\frac{1}{2\sigma^2}x^2}, \tag{A.33}$$

where the standard deviation is $\sigma = \sqrt{N}\epsilon = \sqrt{N}/2$. Hint: In evaluating the logarithm of the probability, keep terms of order $x^2/N$ but neglect higher-order terms in $1/N$ such as $(x/N)^2$.

This constitutes a proof of the central limit theorem for this special case.

*Note that an asymptotic expansion does not mean that the *difference* between the LHS and RHS of Eq. (A.32) becomes arbitrarily close to zero for large $N$. It means that the *ratio* of the RHS to the LHS approaches unity for large $N$. Since both quantities are diverging, their difference can still be large even if their ratio approaches unity.

**Box A.2. Gaussian Integrals** A general Gaussian integral over a single variable has the form

$$I = \int_{-\infty}^{+\infty} dz\, e^{-az^2+bz}. \tag{A.34}$$

It turns out that integrals of this form (and its multi-dimensional generalization) are ubiquitous in physics, so it is handy to know how to carry them out. We begin by 'completing the square' by writing

$$I = \int_{-\infty}^{+\infty} dz\, e^{-a[z-\frac{b}{2a}]^2+\frac{b^2}{4a}}. \tag{A.35}$$

Shifting the dummy variable of integration to $x = z - \frac{b}{2a}$ we have

$$I = e^{\frac{b^2}{4a}} \int_{-\infty}^{+\infty} dx\, e^{-ax^2}. \tag{A.36}$$

It turns out to be easier to consider the square of the integral which we can write as

$$I^2 = \left[e^{\frac{b^2}{4a}}\right]^2 \int_{-\infty}^{+\infty} dxdy\, e^{-a[x^2+y^2]}. \tag{A.37}$$

This two-dimensional integral is readily carried out using polar coordinates $r, \theta$ with $r = \sqrt{x^2+y^2}$

$$I^2 = \left[e^{\frac{b^2}{4a}}\right]^2 \int_{0}^{\infty} 2\pi r dr\, e^{-ar^2}. \tag{A.38}$$

Defining $u = ar^2$ and using $du = 2ardr$ we have

$$I^2 = \frac{\pi}{a}\left[e^{\frac{b^2}{4a}}\right]^2 \int_{0}^{\infty} du\, e^{-u} \tag{A.39}$$

which yields the final result

$$I = \sqrt{\frac{\pi}{a}} e^{\frac{b^2}{4a}}. \tag{A.40}$$

From this we see that the Gaussian probability distribution in Eq. (A.23) is properly normalized to unity.

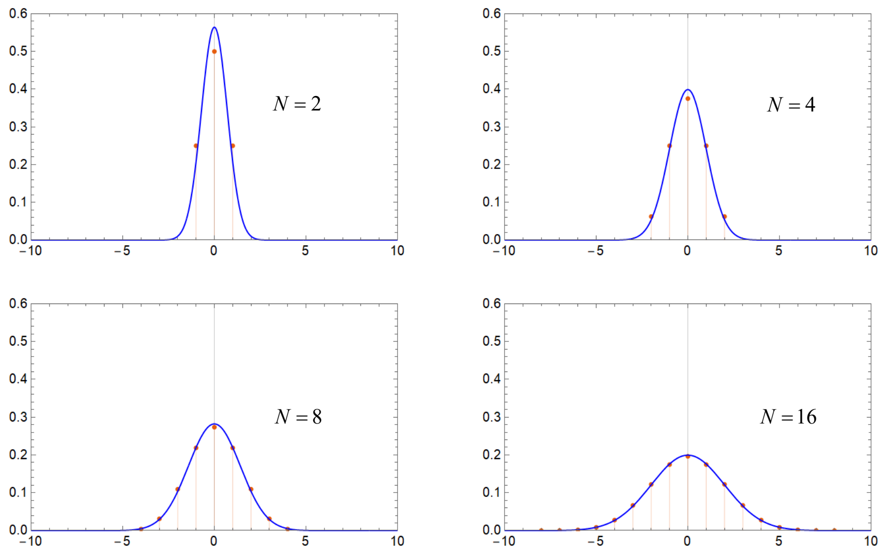

Figure A.2: Dots: Probability distribution for the ending position of random walks of $N$ with step length $\epsilon = 1/2$. The smooth curve is the Gaussian distribution with standard deviation $\sqrt{N}/2$. The Gaussian approximation rapidly becomes very accurate as $N$ increases.

## A.4 Joint Probability Distributions

This section provides background for the discussion of multi-qubit measurements in Sec. 4.2.

Suppose that we have a probability distribution $P(A, B)$ for two random variables, $A$ and $B$. $A$ and $B$ might for example be the measured values of the Pauli $Z$ operator for each two qubits in a quantum computer. Thus for example $P(+1, -1)$ is the probability that the measured value of $Z$ for $q_0$ is $-1$ and the measured value of $Z$ for $q_1$ is $+1$. Altogether there are four possible measurement results for the combined measurement of the two qubits and $P$ obeys the usual sum rule

$$\sum_{A=\pm1} \sum_{B=\pm1} P(A, B) = 1. \tag{A.41}$$

The probability distribution for measurement results on $q_1$ is simply

$$P_1(A) = \sum_{B=\pm1} P(A, B). \tag{A.42}$$

That is, it is the sum of the probabilities of all the independent ways that $q_0$ can be found with measured value $A$ independent of what $B$ is. Notice that it follows immediately from the definition in Eq. (A.42) and the sum rule in Eq. (A.41) that this distribution is properly normalized

$$\sum_{A=\pm 1} P_1(A) = \sum_{A=\pm 1} \sum_{B=\pm 1} P(A,B) = 1. \tag{A.43}$$

Similarly, the probability distribution for measurement results on $q_0$ is

$$P_0(B) = \sum_{A=\pm 1} P(A,B), \tag{A.44}$$

and this too is properly normalized.

## A.4.1 Conditional Probabilities and Bayes Rule

Another useful quantity is the *conditional probability distribution* of $A$ given a fixed value of $B$ which is denoted $P(A|B)$. You might naively think that this is simply given by

$$P(A|B) = P(A,B), \tag{A.45}$$

with $B$ held fixed. This however is not properly normalized. The correctly normalized expression is

$$\begin{aligned} P(A|B) &= \frac{P(A,B)}{\sum_{A=\pm 1} P(A,B)} && \text{(A.46)} \\ &= \frac{P(A,B)}{P_0(B)}. && \text{(A.47)} \end{aligned}$$

From this it follows that

$$P(A,B) = P(A|B)P_0(B). \tag{A.48}$$

Similarly, the probability distribution of $B$ given a fixed value of $A$ is

$$\begin{aligned} P(B|A) &= \frac{P(A,B)}{\sum_{B=\pm 1} P(A,B)} && \text{(A.49)} \\ &= \frac{P(A,B)}{P_1(A)}. && \text{(A.50)} \end{aligned}$$

From this it follows that

$$P(A,B) = P(B|A)P_1(A). \tag{A.51}$$

Equating the above two expressions for $P(A,B)$ yields the very important Bayes rule:

$$P(B|A) = P(A|B)\frac{P_0(B)}{P_1(A)}. \tag{A.52}$$

This is especially powerful in situations (which occur frequently) in which $P(B|A)$ is hard to compute but $P(A|B)$ is easy to compute.

To understand Bayes rule better, let us consider the example discussed in Box 4.6 in which we given the problem of quantifying the information gain from making a measurement. To recap the setup of the problem: Alice randomly gives Bob one copy of one of the following four states

$$\begin{aligned} |\psi_0\rangle &= |0\rangle & \text{(A.53)}\\ |\psi_1\rangle &= |1\rangle & \text{(A.54)}\\ |\psi_2\rangle &= |+\rangle & \text{(A.55)}\\ |\psi_3\rangle &= |-\rangle & \text{(A.56)} \end{aligned}$$

selected with equal probability. Here of course the Shannon entropy of the distribution prior to the measurement is $S_{\text{prior}} = 2\,\text{bits}$. Suppose Bob measures Alice's selected qubit in the $Z$ basis and obtains the result $z = +1$. With this new knowledge he must update the probability distribution from the one prior to the measurement to the new one conditioned on the measurement result. The Born rule tells us immediately what the probability is for the measurement result being the observed $z = +1$ conditioned on the state being $|\psi_j\rangle$

$$\begin{aligned} P(z=+1|\psi_0) &= 1 & \text{(A.57)}\\ P(z=+1|\psi_1) &= 0 & \text{(A.58)}\\ P(z=+1|\psi_2) &= \frac{1}{2} & \text{(A.59)}\\ P(z=+1|\psi_0) &= \frac{1}{2}. & \text{(A.60)} \end{aligned}$$

What we are after however is the reverse, namely the probability that the observed result $z = +1$ was obtained from state $\psi_j$, $P(\psi_j|z = +1)$. This is precisely the situation where Bayes Rule is useful. We have from Eq. (A.52)

$$P(\psi_j|z=+1) = P(z=+1|\psi_j)\frac{P_\psi(\psi_j)}{P_z(z=+1)}, \tag{A.61}$$

where $P_\psi(\psi_j)$ is the prior probability that Alice selects state $\psi_j$. In this case, since Alice chooses each state with equal probability, we have $P_\psi(\psi_j) = \frac{1}{4}$ for all four values of $j$. $P_z(z)$ is the prior probability of obtaining the measurement result $z$. In this case, we find using Eqs. (A.57-A.60)

$$P_z(z=+1) = \sum_{j=0}^{3} P(z=+1|\psi_j)P_\psi(\psi_j) = \left[1+0+\frac{1}{2}+\frac{1}{2}\right]\left(\frac{1}{4}\right) = \frac{1}{2}. \tag{A.62}$$

We see that $P_z(z=+1)$ on the RHS of Eq. (A.62) is just a constant factor needed to fix the normalization of the posterior probability distribution on the LHS.

Combining all these results with Eq. (A.61) yields the results given in Box 4.6

$$\begin{aligned} P(\psi_0|z=+1) &= \frac{1}{2} && \text{(A.63)} \\ P(\psi_1|z=+1) &= 0 && \text{(A.64)} \\ P(\psi_2|z=+1) &= \frac{1}{4} && \text{(A.65)} \\ P(\psi_3|z=+1) &= \frac{1}{4}. && \text{(A.66)} \\ & && \text{(A.67)} \end{aligned}$$

As noted in Box 4.6, this probability distribution has Shannon entropy of (3/2) bits. Thus the information gain Bob receives from his measurement is

$$I = S_{\text{prior}} - S_{\text{post}} = \left(2 - \frac{3}{2}\right) \text{ bits} = \frac{1}{2} \text{ bit.}$$

# Appendix B

# Formal Definition of a Vector Space

In classical physics and engineering, we generally work with simple three-component position, velocity and acceleration vectors consisting of a triple of real numbers. The concept of a vector space is however much more general than this. Strictly speaking, we should refer to a 'vector space over a field.' Here are the properties:

We need a field $S$ (e.g., a set of objects like the real numbers or the complex numbers on which two operations, multiplication and addition are defined). The elements of the field are called scalars.

The vectors are a set of objects $\{V\}$ with two operations:

- addition, so that for $\vec{v}_1, \vec{v}_2 \in \{V\}$, $\vec{v}_1 + \vec{v}_2$ is also in $\{V\}$ (*closure under addition*).
- multiplication by a scalar. If $s \in S$ then $s\vec{v}_1 \in \{V\}$ (*closure under scalar multiplication*).

The following eight properties must be satsified:

1. $(\vec{v}_1 + \vec{v}_2) + \vec{v}_3 = \vec{v}_1 + (\vec{v}_2 + \vec{v}_3)$ (addition is associative)
2. $\vec{v}_1 + \vec{v}_2 = \vec{v}_2 + \vec{v}_1$ (addition is commutative)
3. $\exists\, \vec{0} \in \{V\}$ such that $\vec{0} + \vec{v} = \vec{v}$ (additive identity exists)
4. $\forall\, \vec{v} \in \{V\}, \exists - \vec{v} \in \{V\}$ such that $\vec{v} + (-\vec{v}) = \vec{0}$. (additive inverse exists)

5. $s(\vec{v}_1 + \vec{v}_2) = s\vec{v}_1 + s\vec{v}_2$ (1st distribution law)
6. $(s_1 + s_2)\vec{v} = s_1\vec{v} + s_2\vec{v}$ (2nd distribution law)
7. $s_1(s_2\vec{v}) = (s_1 s_2)\vec{v}$
8. $1(\vec{v}) = \vec{v}$ (where 1 is the multiplicative identity of the scalar field)

## B.1 Basis Vectors and the Dimension of a Vector Space

The *dimension* $D$ of a vector space is the maximum number[1] of linearly independent vectors. Suppose that we have a set of vectors $Q = \{B_1, B_2, \ldots, B_N\}$ and we consider the function

$$F(s_1, s_2, \ldots, s_N) = \sum_{j=1}^{N} s_j B_j. \tag{B.1}$$

The set $Q$ of vectors is by definition *linearly independent* if the only solution to the equation $F(s_1, s_2, \ldots, s_N) = 0$ is that all of its arguments vanish $(s_1, s_2, \ldots, s_N) = (0, 0, \ldots, 0)$. If $N > D$ there are too many vectors to all be pointing in orthogonal directions and some of the vectors are redundant in the sense that they can be expressed as linear combinations of other vectors. In this case $F = 0$ has solutions for non-zero arguments of $F$.

A *basis* for a vector space of dimension $D$ is a set of vectors $Q = \{B_1, B_2, \ldots, B_D\}$ which *span* the space. That is every vector $V$ in the space can be written as a (unique) linear combination of the basis vectors

$$V = \sum_{j=1}^{D} b_j B_j \tag{B.2}$$

by choosing appropriate values for the set of scalars (coefficients) $(b_1, b_2, \ldots, b_D)$. This list of coefficients constitutes a representation of the abstract vector $V$ in this basis $B$.

[1]We consider here only the case where the Hilbert space dimension is finite (or at least countable).

As a familiar example, the unit vectors

$$\hat{i} = (1,0,0) \tag{B.3}$$

$$\hat{j} = (0,1,0) \tag{B.4}$$

$$\hat{k} = (0,0,1) \tag{B.5}$$

can be used to form any position vector in ordinary three-dimensional space

$$\vec{r} = (x,y,z) = x\hat{i} + y\hat{j} + z\hat{k}. \tag{B.6}$$

These Cartesian coordinate basis vectors happen to be orthogonal (formally defined below) because they are perpendicular in the usual geometric sense. However a basis set is not required to be orthogonal, only complete.

## B.2 Inner Products and Norms

For the ordinary vectors like positions and displacements (differences of position vectors) that we are used to the concept of the length (or 'norm') of a vector defined in terms of Pythagorean theorem. If

$$\vec{A} = (A_x, A_y, A_z), \tag{B.7}$$

then the norm is

$$|\vec{A}| = \sqrt{A_x^2 + A_y^2 + A_z^2}. \tag{B.8}$$

The general definition of a norm for a vector space is (more or less) any mapping from the vectors to the non-negative real numbers satisfying the triangle inequality $|\vec{A} + \vec{B}| \leq |\vec{A}| + |\vec{B}|$. It is important to note that a given vector space need not have a norm defined.

For ordinary vectors we are used to the dot product

$$\vec{A} \cdot \vec{B} = a_x B_x + A_y B_y + A_z B_z = |\vec{A}||\vec{B}| \cos\theta_{AB}, \tag{B.9}$$

where $\theta_{AB}$ is the angle between the two vectors. The dot product is a specific example of the general concept of an *inner product*. An inner product of two abstract vectors $V_1$ and $V_2$ is a mapping onto a scalar $s$, often denoted

$$(V_1, V_2) = s. \tag{B.10}$$

For the case where the vector space is defined over the field of complex numbers, an inner product satisfies the following requirements:

- Linearity in the second argument

$$(V_1, aV_2) = as, \tag{B.11}$$
$$(V_1, V_2 + V_3) = (V_1, V_2) + (V_1, V_3), \tag{B.12}$$

- Anti-linearity (conjugate linearity) in the first argument

$$(aV_1, V_2) = a^*(V_1, V_2), \tag{B.13}$$
$$(V_1, V_2) = (V_2, V_1)^*, \tag{B.14}$$

- Positive semi-definite

$$(V_1, V_1) \geq 0 \tag{B.15}$$

  and

$$(V_1, V_1) = 0 \implies V_1 = \vec{0}. \tag{B.16}$$

  [Note the first zero above is the scalar zero and the arrow is placed on the second zero to make clear that this is the null vector (additive zero vector) not the scalar zero.]

Two vectors are defined to be *orthogonal* if their inner product vanishes.

**Exercise B.1.** Prove that the usual dot product for real three-dimensional vectors satisfies the definition of an inner product.

The Pythagorean norm defined above for ordinary real vectors is thus related to the inner product of the vector with itself

$$|\vec{A}| = \sqrt{\vec{A} \cdot \vec{A}}. \tag{B.17}$$

**Exercise B.2.** Prove that for a general complex vector space, $\sqrt{(V,V)}$ satisfies the definition of a norm.

In describing qubit states we will deal with two component complex valued vectors of the form

$$\Psi_1 = (\alpha_1, \beta_1), \tag{B.18}$$
$$\Psi_2 = (\alpha_2, \beta_2). \tag{B.19}$$

The standard inner product and norm for such vectors are defined in Ex. B.3.

**Exercise B.3.** Prove that the following generalization of the dot product to complex vectors satisfies the definition of an inner product

$$(\Psi_1, \Psi_2) = \alpha_1^* \alpha_2 + \beta_1^* \beta_2. \tag{B.20}$$

and prove that

$$\sqrt{(\Psi_1, \Psi_1)} = \sqrt{\alpha_1^* \alpha_1 + \beta_1^* \beta_1}. \tag{B.21}$$

satisfies the definition of a norm.

## B.3 Dirac Notation, Outer Products and Operators for Single and Multiple Qubits

Physicists generally use a notation for complex state vectors and their inner products that was developed by Paul Adrian Maurice Dirac. In this notation the complex vector in Eq. (B.18) is represented as column vector instead of a row vector using the 'bracket' notation

$$|\Psi_1\rangle = \begin{pmatrix} \alpha_1 \\ \beta_1 \end{pmatrix}, \tag{B.22}$$

where $|\Psi_1\rangle$ is referred to as a 'ket' vector and the inner product is represented by

$$(\Psi_1, \Psi_2) = \langle \Psi_1 | \Psi_2 \rangle, \tag{B.23}$$

where the 'dual vector'

$$\langle \Psi_1 | = (\alpha_1^*, \beta_1^*)\,, \tag{B.24}$$

is a row vector representing the conjugate transpose of the column vector. In this notation, the inner product is computed using the rules of matrix multiplication

$$\langle \Psi_1 | \Psi_2 \rangle = (\alpha_1^*, \beta_1^*) \begin{pmatrix} \alpha_2 \\ \beta_2 \end{pmatrix} = \alpha_1^* \alpha_2 + \beta_1^* \beta_2. \tag{B.25}$$

The Dirac notation is also very convenient for defining the *outer product* of two vectors which, as described in Chapter 3 is a linear operator that maps the vector space back onto itself (i.e. maps vectors onto other vectors)

$$\mathcal{O} = |\Psi_2\rangle\langle\Psi_1|. \tag{B.26}$$

Applying this operator to a vector $|\Psi_3\rangle$ yields

$$\mathcal{O}|\Psi_3\rangle = (|\Psi_2\rangle\langle\Psi_1|)\,|\Psi_3\rangle = |\Psi_2\rangle\langle\Psi_1|\Psi_3\rangle = |\Psi_2\rangle\,(\langle\Psi_1|\Psi_3\rangle) = s_{13}|\Psi_2\rangle, \tag{B.27}$$

where the scalar $s_{13}$ is given by the inner product (also known in quantum parlance as the 'overlap')

$$s_{13} = \langle\Psi_1|\Psi_3\rangle. \tag{B.28}$$

Subsituting the definitions of of the bra and ket vectors in Eq. (B.26) we see that the abstract operator can be represented as a matrix

$$\mathcal{O} = \begin{pmatrix} \alpha_2 \\ \beta_2 \end{pmatrix} (\alpha_1^*, \beta_1^*) = \begin{pmatrix} \alpha_2\alpha_1^* & \alpha_2\beta_1^* \\ \beta_2\alpha_1^* & \beta_2\beta_1^* \end{pmatrix}. \tag{B.29}$$

By simply switching the order of the row and column vector from that in Eq. (B.25), the rules of matrix multiplication give us a matrix instead of a scalar!

**Exercise B.4.** Use the matrix representation of $\mathcal{O}$ in Eq. (B.29) and apply it to the column vector representation of $|\Psi_3\rangle$

$$|\Psi_3\rangle = \begin{pmatrix} \gamma \\ \delta \end{pmatrix}$$

to verify the last equality in Eq. (B.27.)

Recall that the adjoint of the product of two matrices is the product of the adjoints in reverse order

$$(AB)^\dagger = B^\dagger A^\dagger. \tag{B.30}$$

It works the same way in the Dirac notation for the outer product of two vectors that forms an operator. Thus the adjoint of the operator in Eq. (B.26) is simply

$$\mathcal{O}^\dagger = |\Psi_1\rangle\langle\Psi_2|. \tag{B.31}$$

To see that this is true, it is best to work with the representation in Eq. (B.29)

$$\begin{aligned}
\mathcal{O}^\dagger &= \left[\begin{pmatrix} \alpha_2 \\ \beta_2 \end{pmatrix} (\alpha_1^*, \beta_1^*)\right]^\dagger = \left[\begin{pmatrix} \alpha_2\alpha_1^* & \alpha_2\beta_1^* \\ \beta_2\alpha_1^* & \beta_2\beta_1^* \end{pmatrix}\right]^\dagger, && \text{(B.32)}\\
&= \begin{pmatrix} \alpha_1 \\ \beta_1 \end{pmatrix} (\alpha_2^*, \beta_2^*) = \begin{pmatrix} \alpha_1\alpha_2^* & \alpha_1\beta_2^* \\ \beta_1\alpha_2^* & \beta_1\beta_2^* \end{pmatrix}, && \text{(B.33)}\\
&= |\Psi_1\rangle\langle\Psi_2|. && \text{(B.34)}
\end{aligned}$$

## Tensor Products and Multiqubit States

The so-called tensor product of a pair of $2 \times 2$ matrices produces a $4 \times 4$ matrix

$$\begin{aligned}
A \otimes B &= \begin{pmatrix} A_{11}[B] & A_{12}[B] \\ A_{21}[B] & A_{22}[B] \end{pmatrix} && \text{(B.35)}\\
&= \begin{pmatrix} A_{11}\begin{pmatrix} B_{11} & B_{12} \\ B_{21} & B_{22} \end{pmatrix} & A_{12}\begin{pmatrix} B_{11} & B_{12} \\ B_{21} & B_{22} \end{pmatrix} \\ A_{21}\begin{pmatrix} B_{11} & B_{12} \\ B_{21} & B_{22} \end{pmatrix} & A_{22}\begin{pmatrix} B_{11} & B_{12} \\ B_{21} & B_{22} \end{pmatrix} \end{pmatrix}. && \text{(B.36)}
\end{aligned}$$

Such tensor products appear when dealing with the Hilbert space of two qubits where the operator $A$ acts on one qubit and operator $B$ acts on the other. This Hilbert space has dimension four and the states are represented by column vectors of length four

$$|\psi\rangle = \begin{pmatrix} \psi_0 \\ \psi_1 \\ \psi_2 \\ \psi_3 \end{pmatrix}. \qquad \text{(B.37)}$$

For the Hilbert space of $n$ qubits, we will use the following Dirac notation for state vectors and their duals in the computational basis

$$\begin{aligned}
|\psi\rangle &= |b_{n-1} \dots b_2 b_1 b_0\rangle && \text{(B.38)}\\
\langle\psi| &= \langle b_{n-1} \dots b_2 b_1 b_0| && \text{(B.39)}
\end{aligned}$$

in which the qubits are numbered from 0 to $n-1$ and their values $b_j \in \{0, 1\}$ are ordered from right to left. (Note that we maintain this same label ordering in the dual vector.) The computational basis is simply the tensor product of the computational basis of the individual qubits (illustrated here for the case of two qubits)

$$\begin{aligned}
|00\rangle &= |0\rangle \otimes |0\rangle = \begin{pmatrix} 1 \\ 0 \end{pmatrix} \otimes \begin{pmatrix} 1 \\ 0 \end{pmatrix} = \begin{pmatrix} 1 \\ 0 \\ 0 \\ 0 \end{pmatrix} && \text{(B.40)} \\
|01\rangle &= |0\rangle \otimes |1\rangle = \begin{pmatrix} 1 \\ 0 \end{pmatrix} \otimes \begin{pmatrix} 0 \\ 1 \end{pmatrix} = \begin{pmatrix} 0 \\ 1 \\ 0 \\ 0 \end{pmatrix} && \text{(B.41)} \\
|10\rangle &= |1\rangle \otimes |0\rangle = \begin{pmatrix} 0 \\ 1 \end{pmatrix} \otimes \begin{pmatrix} 1 \\ 0 \end{pmatrix} = \begin{pmatrix} 0 \\ 0 \\ 1 \\ 0 \end{pmatrix} && \text{(B.42)} \\
|11\rangle &= |1\rangle \otimes |1\rangle = \begin{pmatrix} 0 \\ 1 \end{pmatrix} \otimes \begin{pmatrix} 0 \\ 1 \end{pmatrix} = \begin{pmatrix} 0 \\ 0 \\ 0 \\ 1 \end{pmatrix}. && \text{(B.43)}
\end{aligned}$$

Notice that if we label the ordinal positions in the column vector starting with 0 at the top and ending with 3 as in Eq. (B.37), then the binary representation of the position of the entry containing 1 gives the state of the two qubits in the computational basis. For example $|11\rangle$ corresponds to the binary representation of the number 3 which in turn corresponds to the location of the entry 1 being at the bottom of the column vector in Eq. (B.43)

We can also write the dual vectors associated with the above two-qubit state vectors. For example the dual of the vector in Eq. (B.41) is

$$\langle 01| = \langle 0| \otimes \langle 1| = \begin{pmatrix} 1 & 0 \end{pmatrix} \otimes \begin{pmatrix} 0 & 1 \end{pmatrix} = \begin{pmatrix} 0 & 1 & 0 & 0 \end{pmatrix}. \qquad \text{(B.44)}$$

Recall from Eq. (B.30) for ordinary products of matrices we need to reverse the order of the matrices when forming the transpose. However in forming the dual of the tensor product $|0\rangle \otimes |1\rangle$, we do *not* reverse the order of the two terms in the tensor product. This is because of our convention of keeping the bit order the same when writing the dual of $|01\rangle$ as $\langle 01|$ rather than $\langle 10|$.

As examples of operators acting on this Hilbert space consider the joint

Pauli operators

$$Z \otimes X = \begin{pmatrix} 0 & +1 & 0 & 0 \\ +1 & 0 & 0 & 0 \\ 0 & 0 & 0 & -1 \\ 0 & 0 & -1 & 0 \end{pmatrix}, \tag{B.45}$$

and

$$X \otimes Z = \begin{pmatrix} 0 & 0 & +1 & 0 \\ 0 & 0 & 0 & -1 \\ +1 & 0 & 0 & 0 \\ 0 & -1 & 0 & 0 \end{pmatrix}. \tag{B.46}$$

It is straightforward to verify that (for example)

$$(Z \otimes X)|10\rangle = \begin{pmatrix} 0 & +1 & 0 & 0 \\ +1 & 0 & 0 & 0 \\ 0 & 0 & 0 & -1 \\ 0 & 0 & -1 & 0 \end{pmatrix} \begin{pmatrix} 0 \\ 0 \\ 1 \\ 0 \end{pmatrix} = - \begin{pmatrix} 0 \\ 0 \\ 0 \\ 1 \end{pmatrix} \tag{B.47}$$

is equivalent to (i.e., correctly represents)

$$\begin{aligned} (Z \otimes X)|10\rangle &= (Z|1\rangle) \otimes (X|0\rangle) && \text{(B.48)} \\ &= (-|1\rangle) \otimes (|1\rangle) && \text{(B.49)} \\ &= -|11\rangle. && \text{(B.50)} \end{aligned}$$

Similarly

$$(X \otimes Z)|10\rangle = \begin{pmatrix} 0 & 0 & +1 & 0 \\ 0 & 0 & 0 & -1 \\ +1 & 0 & 0 & 0 \\ 0 & -1 & 0 & 0 \end{pmatrix} \begin{pmatrix} 0 \\ 0 \\ 1 \\ 0 \end{pmatrix} = \begin{pmatrix} 1 \\ 0 \\ 0 \\ 0 \end{pmatrix}, \tag{B.51}$$

is equivalent to

$$\begin{aligned} (X \otimes Z)|10\rangle &= (X|1\rangle) \otimes (Z|0\rangle) && \text{(B.52)} \\ &= (|0\rangle) \otimes (+|0\rangle) && \text{(B.53)} \\ &= +|00\rangle. && \text{(B.54)} \end{aligned}$$

The above examples show us that the tensor product of two operators is represented by a $4 \times 4$ matrix that can act on the column vector representing the tensor product of two single qubit states. But we also see that this is exactly equivalent to each operator acting separately on their respective qubits and then taking the tensor product of the resulting state vectors. If we want an operator that acts on only qubit $q_0$ we simply tensor it with the identity acting on $q_1$. For example,

$$X_0 = I \otimes X = \begin{pmatrix} 0 & 1 & 0 & 0 \\ 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 1 & 0 \end{pmatrix}, \tag{B.55}$$

whereas if we want the operator to act only on $q_1$ we should use

$$X_0 = X \otimes I = \begin{pmatrix} 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \\ 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \end{pmatrix}. \tag{B.56}$$

It is important to note that the tensor product of two Pauli matrices is not the most general form of two-qubit operator. For example, the CNOT gate shown in Eq. (4.19) is a sum of four different products of Pauli operators.

## B.4 The Vector Space of Bit Strings

Computer and information science deals with bit strings and it turns out that these form a vector space over a field. To see this, consider the set of all possible bit strings $\vec{x} = (x_{n-1}x_{n-2}\dots x_2x_1x_0)$ of length $n$. Since every element in the vector is a bit whose only allowed values are 0 and 1, we define addition via

$$\vec{x} \oplus \vec{y} \equiv \vec{x} + \vec{y} \mod 2, \tag{B.57}$$

where the mod 2 operation is performed bitwise. Strangely, this means that every vector is its own additive inverse

$$\vec{x} \oplus \vec{x} = \vec{0}. \tag{B.58}$$

Similarly the only allowed scalars are $s = 0, 1$ because only for these values are vectors of the form $s\vec{x}$ in the space of bit strings. The set of scalars $\{0, 1\}$

together with the operations of ordinary multiplication and addition mod 2, constitute a field. This binary field is traditionally denoted $\mathbb{F}_2$. Given these facts, it is straightforward to verify that the criteria required for the set of bit strings of length $n$ to be a vector space over a field are all satisfied.

We can define an inner product on this vector space

$$\vec{x}.\vec{y} = \left(\sum_{j=0}^{n-1} x_j y_j\right) \mod 2 = (\vec{x}\cdot\vec{y}) \mod 2, \tag{B.59}$$

where $\vec{x}\cdot\vec{y}$ is the ordinary Euclidean dot product. (The inner product has to be a scalar and there are only two allowed scalars which is why the mod 2 arithmetic is required in the inner product.) The 'length' of a vector $L^2 = L = \vec{x}.\vec{x}$ is thus the parity of the number non-zero bits in the string. $L = 0$ if the vector contains an even number of 1's and $L = 1$ if the vector contains an odd number of 1's.

This notion of 'length' does not give a very complete notion of the distance between two vectors. To remedy this one can define the notion of the *Hamming distance* between two vectors

$$d_{\mathrm{H}}(\vec{x},\vec{y}) = \sum_{j=0}^{n-1}(x_j \oplus y_j). \tag{B.60}$$

Because $x_j \oplus y_j$ is zero if the two bits agree and one if they differ, the Hamming distance is the total number of instances where the bit strings differ. Equivalently it is the total number of bits in $\vec{x}$ that would need to be flipped to convert $\vec{x}$ into $\vec{y}$.

The notion of Hamming distance is very important in classical error correction, because the Hamming distance between a code word (a bit string in the code space) and a word that has been corrupted by errors (bit flips) is equal to the number of bitflip errors. These various notions of length and distance are not to be confused with the 'length' $n$ of a bit string in the ordinary sense of the number of bits in the bit string vectors, which is the dimension of the vector space.

# Appendix C

# Handy Mathematical Identities

## Useful Information about the Mathematical Representation of Qubit States and Operations

Standard basis states:

$$\begin{aligned}
|0\rangle = |\uparrow\rangle &= \begin{pmatrix} 1 \\ 0 \end{pmatrix} \\
|1\rangle = |\downarrow\rangle &= \begin{pmatrix} 0 \\ 1 \end{pmatrix}.
\end{aligned}$$

The representation of the basis states in the $\hat{n}$ basis in terms of the standard basis states is

$$\begin{aligned}
|+\hat{n}\rangle &= \cos\left(\frac{\theta}{2}\right)|0\rangle + e^{i\varphi}\sin\left(\frac{\theta}{2}\right)|1\rangle \\
|-\hat{n}\rangle &= \sin\left(\frac{\theta}{2}\right)|0\rangle - e^{i\varphi}\cos\left(\frac{\theta}{2}\right)|1\rangle
\end{aligned}$$

Standard basis states:

$$\begin{aligned}
|0\rangle = |\uparrow\rangle &= \begin{pmatrix} 1 \\ 0 \end{pmatrix} \\
|1\rangle = |\downarrow\rangle &= \begin{pmatrix} 0 \\ 1 \end{pmatrix}.
\end{aligned}$$

The representation of the basis states in the $\hat{n}$ basis in terms of the standard basis states is

$$\begin{aligned} |+\hat{n}\rangle &= \cos\left(\frac{\theta}{2}\right)|0\rangle + e^{i\varphi}\sin\left(\frac{\theta}{2}\right)|1\rangle \\ |-\hat{n}\rangle &= \sin\left(\frac{\theta}{2}\right)|0\rangle - e^{i\varphi}\cos\left(\frac{\theta}{2}\right)|1\rangle \end{aligned}$$

where $\hat{n} = (\sin\theta\cos\varphi, \sin\theta\sin\varphi, \cos\theta)$.

Pauli matrices:

$$\begin{aligned} X = \sigma^x &= \begin{pmatrix} 0 & +1 \\ +1 & 0 \end{pmatrix} \\ Y = \sigma^y &= \begin{pmatrix} 0 & -i \\ +i & 0 \end{pmatrix} \\ Z = \sigma^z &= \begin{pmatrix} +1 & 0 \\ 0 & -1 \end{pmatrix} \\ I = \sigma_0 &= \begin{pmatrix} +1 & 0 \\ 0 & +1 \end{pmatrix}. \end{aligned}$$

Trace of Pauli matrices

$$\begin{aligned} \operatorname{Tr} X = \operatorname{Tr} Y = \operatorname{Tr} Z = 0 \\ \operatorname{Tr} I = 2. \end{aligned}$$

Products of Pauli Matrices

$$\begin{aligned} X^2 &= Y^2 = Z^2 = I \\ XY &= -YX = iZ \\ YZ &= -ZY = iX \\ ZX &= -XZ = iY \\ XYZ &= iI. \end{aligned}$$

Commutators of Pauli matrices:

$$\begin{aligned} [X,Y] = XY - YX = 2iZ \\ [Y,Z] = YZ - ZY = 2iX \\ [Z,X] = ZX - XZ = 2iY. \end{aligned}$$

Anticommutators of Pauli matrices

$$\begin{aligned}\{X,Y\} &= XY+YX=0\\ \{Y,Z\} &= YZ+ZY=0\\ \{Z,X\} &= ZX+XZ=0.\end{aligned}$$

Eigenstates of $\sigma^x$:

$$|\pm\rangle = \frac{1}{\sqrt{2}}\left[|0\rangle \pm |1\rangle\right].$$

Eigenstates of $\sigma^y$:

$$|\pm i\rangle = \frac{1}{\sqrt{2}}\left[|0\rangle \pm i|1\rangle\right].$$

Euler-Pauli Identity: If $A^2$ is the identity operator $I$ (for any dimension Hilbert space) then for real $\theta$, $e^{i\theta A} = [\cos\theta]I + [i\sin\theta]A$.

Rotation of qubit on Bloch sphere by an angle $\chi$ around the $\hat{\omega}$ axis (recall the right-hand rule)

$$R_{\hat{\omega}}(\chi) = e^{-i\frac{\chi}{2}\hat{\omega}\cdot\vec{\sigma}} = \cos(\chi/2)\,\hat{I} - i\sin(\chi/2)\,\hat{\omega}\cdot\vec{\sigma}.$$

Standard right-to-left ordering of multi-qubit states in the computational basis $|q_{n-1}, q_{n-2}, \ldots, q_2, q_1, q_0\rangle$. Generic two-qubit state is:

$$|\Psi\rangle = \psi_{00}|00\rangle + \psi_{01}|01\rangle + \psi_{10}|10\rangle + \psi_{11}|11\rangle = \begin{pmatrix}\psi_{00}\\ \psi_{01}\\ \psi_{10}\\ \psi_{11}\end{pmatrix}.$$

In addition to the standard orthonormal computational basis states for two qubits, $\{|00\rangle, |01\rangle, |10\rangle, |11\rangle\}$, another commonly used orthonormal basis for two qubits are the so-called Bell states:

$$\begin{aligned}|B_0\rangle &= \frac{1}{\sqrt{2}}\left[|01\rangle - |10\rangle\right]\\ |B_1\rangle &= \frac{1}{\sqrt{2}}\left[|01\rangle + |10\rangle\right]\\ |B_2\rangle &= \frac{1}{\sqrt{2}}\left[|00\rangle - |11\rangle\right]\\ |B_3\rangle &= \frac{1}{\sqrt{2}}\left[|00\rangle + |11\rangle\right].\end{aligned}$$

Handy trig identities:

$$\begin{aligned}
\cos(\xi) &= \cos^2\left(\frac{\xi}{2}\right) - \sin^2\left(\frac{\xi}{2}\right) \\
\sin(\xi) &= 2\sin\left(\frac{\xi}{2}\right)\cos\left(\frac{\xi}{2}\right) \\
\sin\left(\frac{\xi}{2}\right) &= -\sin\left(\frac{\xi}{2}\right)\cos(\xi) + \cos\left(\frac{\xi}{2}\right)\sin(\xi) \\
\cos\left(\frac{\xi}{2}\right) &= \sin\left(\frac{\xi}{2}\right)\sin(\xi) + \cos\left(\frac{\xi}{2}\right)\cos(\xi) \\
\sin(\pi/4) &= \cos(\pi/4) = 1/\sqrt{2}.
\end{aligned}$$

# Appendix D

# Suggestions for Projects

The TF's and the Peer Tutors as well as members of the Yale Undergraduate Quantum Computing Club can help with selecting and carrying out your project. The goal is to learn something not covered in class and produce a $\sim$ 10 page pedagogical write up that will allow other students to learn the topic as well.

## D.1 Gates and Instruction Sets

1. Clifford group; Clifford operations produce entanglement but are easy to simulate classically by updating the stabilizer set.

2. Clifford gates (X,Y,Z,H,CNOT) plus the T gate is universal; What is the circuit depth required to synthesize an arbitrary single-qubit rotation to precision $\epsilon$?

3. Toffoli plus Hadamard gate set is universal:

   http:arxiv.org/abs/quant-ph/0301040v1

4. In classical reversible computing a three bit gate is required for universality (e.g. Toffoli or Fredkin). Single bit operations plus reversible two-qubit gates (e.g. CNOT) are insufficient for universal computation because it is impossible to synthesize a Toffoli (CCNOT )or Fredkin (CSWAP) with classical CNOTs. Quantum mechanically it is possible. See: Chau and Wilczek, *Phys. Rev. Lett.* **75**, 748 (1995); Smolin and Divincenzo, *Phys. Rev. A* **53**, 2855 (1996).

5. Quantum Random Access Memory (QRAM)
   Used to query a database with a superposition of addresses. Numerous applications.
   For example, here are some references that describe how classical data may be encoded in the amplitudes of a quantum state.

   (a) The basic idea was first presented in https://arxiv.org/pdf/quant-ph/0208112.pdf. However, in that work, they don't use QRAM. Rather, they assume a restriction that the data is efficiently integrable, which obviates the need for QRAM.

   (b) In the general case, we can use QRAM to encode data in the amplitudes, provided that the QRAM also stores partial sums of the form $\sum_{i=n}^{m} |x_i|^2$, where $x$ is the data vector. This is described in https:arxiv.org/abs/1812.00954 (in the vicinity of Eq. (8)), in https:arxiv.org/pdf/1607.05256.pdf (proposition 3.3.5), and in https:arxiv.org/pdf/1603.08675.pdf (Theorem A1).

   (c) In situations where the data cannot be efficiently integrated, and where the partial sums are not available, then it is still possible to prepare the amplitude encoding probabilistically using QRAM. This is described in https:arxiv.org/pdf/1804.00281.pdf (in the vicinity of Eq. (3)). This approach is not always efficient, as the success probability can be small.

6. Alternatives to the Circuit Model of Quantum Computing

   (a) Cluster state quantum computing uses only an initial entangled state plus measurement and feed-forward

   (b) Adiabatic quantum computation (for students with enough quantum mechanics background to understand the adiabatic theorem for state evolution under a slowly varying Hamiltonian)

## D.2 Efficient Representation of Quantum States

1. Matrix Product States; PEPS; Tensor Product Methods; their use in VQE and QAOA

## D.3 Algorithms

We will cover some of the algorithms described in Kaye, Laflamme and Mosca (*An Introduction to Quantum Computing*) but not all, so this is a useful reference. Rieffel and Polak (*Quantum Computing: A Gentle Introduction*) is also useful.

The review by Ashley Montaro (doi:10.1038/npjqi.2015.23) on quantum algorithms is a good starting point. This and other papers can be found in "Papers for Projects" folder in the files section of Canvas.

The following websites maintain a large list of quantum algorithms and protocols

https://quantumalgorithmzoo.org/
https://wiki.veriqloud.fr/index.php?title=Protocol_Library

Here are a few suggestions:

1. Quantum Walks (quantum versions of statistical random walks)
2. Accelerated Grover Search? Korepin https://arxiv.org/abs/2102.01783
3. Oblivious Amplification
4. Amplitude Estimation and Quantum Counting
5. Phase estimation (may cover some aspects of this in class but you could go into more depth or could experiment with executing in on the IBM machines)
6. Quantum Fourier Transform (We will do this in class but you could do a more detailed comparison with the classical Fast Fourier Transform Algorithm, experiment with executing the QFT on the IBM machines.)
7. Shor's factoring algorithm
8. Discrete Logarithm algorithm
9. Hidden Subgroup Problem
10. Graph Isomorphism Problem
11. HHL Algorithm for Linear Algebra

12. Quantum Machine Learning

13. Quantum Singular Value transformation (See the colloquium talk by Isaac Chuang:

    `https://www.youtube.com/watch?v=NZ5PmIaJ5IE`

    )

14. Variational Quantum Estimation (VQE) (will cover in class but you could apply this to a specific problem using the IBM Q system)

15. Quantum Approximate Optimization Algorithm (QAOA) (closely related to VQE; will cover in class but you could apply this to a specific problem using the IBM Q system)

16. Quantum secret sharing

17. Quantum bit commitment

18. Quantum multi-party function evaluation

## D.4 Quantum Error Correction and Fault Tolerance

1. Steane vs. Knill error correction and leakage errors

2. Concatenation of codes and the code capacity threshold

3. Families of surface codes

4. Entanglement purification and distillation

5. Magic state injection

6. Easton-Knill theorem on transversal gates and fault tolerance

7. Quantum Low-Density Parity Check (LDPC) Codes; CSS codes;

## D.5 Quantum Communication and Security

1. Bell Inequalities (discussed in course notes but not in class); CHSH test, Mermin 3 qubit test, Experiments on Loophole-free Bell tests; Contextuality in quantum mechanics; 'magic states'

2. Quantum encryption using entanglement (BB84 which does not use entanglement will be covered in class)

   (a) 'Quantum cryptography based on Bell's theorem,' Artur K. Ekert, *Phys. Rev. Lett.* **67**, 661 (1991).

   (b) 'Quantum Cryptography: Public Key Distribution and Coin Tossing,' Charles H. Bennett and Gilles Brassard, arxiv.org/abs/2003.06557.

   (c) 'Entanglement-based quantum communication over 144km,' R. Ursin et al. (Zeilinger group), doi:10.1038/nphys629

3. Quantum channel capacity; hashing bound

## D.6 Quantum Complexity Classes

1. Sampling Complexity

2. Quantum Supremacy Claim: Cross entropy benchmark and Heavy Output Sampling; may cover this in class but you could do experiments on the IBM Q; How good is the maximum entropy assumption for random unitaries?

## D.7 Quantum Hardware Platforms

1. Ion traps

2. Rydberg atom arrays

3. Continuous variable quantum information processing (using harmonic oscillators)

4. Superconducting qubits

(a) circuit QED (physics background needed)

(b) Prof. Jens Koch at Northwestern
(https://sites.northwestern.edu/koch/)
has volunteered to advise a project on using his scQubits Python platform to solve the Schrödinger equation for the energy level structure of superconducting qubits

# Bibliography

[1] Scarani, Valerio and Iblisdir, Sofyan and Gisin, Nicolas and Acín, Antonio, 'Quantum cloning,' *Rev. Mod. Phys.* **77**, 1225 (2005).

[2] Bennett, Charles H. and Wiesner, Stephen J., 'Communication via one- and two-particle operators on Einstein-Podolsky-Rosen states,' *Phys. Rev. Lett.* **69**, 2881-2884 (1992).

[3] A. Einstein and B. Podolsky and N. Rosen, 'Can quantum-mechanical description of physical reality be considered complete?', *Phys. Rev.* **47,** 777-780 (1935).

[4] J. S. Bell, 'On the Einstein Podolsky Rosen paradox,' *Physics* **1**, 195–200 (1964).

[5] J. F. Clauser, M. A. Horne, A. Shimony, and R. A. Holt, 'Proposed experiment to test local hidden-variable theories,' *Phys. Rev. Lett.* **23**, 880–884, (1969).

[6] C. H. Bennett, G. Brassard, C. Crépeau, R. Jozsa, A. Peres, and W. K. Wootters, 'Teleporting an unknown quantum state via dual classical and Einstein-Podolsky-Rosen channels,' *Phys. Rev. Lett.* **70**, 1895–1899 (1993).