An Undergraduate Course on Quantum Computing

Peter Young
# Contents

<table>
<thead>
<tr>
<th>Section</th>
<th>Title</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>Preface</td>
<td></td>
<td>vii</td>
</tr>
<tr>
<td><strong>1</strong></td>
<td>The Strange World of Quantum Mechanics</td>
<td>1</td>
</tr>
<tr>
<td>1.1</td>
<td>Introduction</td>
<td>1</td>
</tr>
<tr>
<td>1.2</td>
<td>The Two-Slit Experiment</td>
<td>2</td>
</tr>
<tr>
<td>1.3</td>
<td>Stern-Gerlach Experiment</td>
<td>4</td>
</tr>
<tr>
<td>1.4</td>
<td>Photons</td>
<td>6</td>
</tr>
<tr>
<td><strong>2</strong></td>
<td>Review of Linear Algebra</td>
<td>9</td>
</tr>
<tr>
<td>2.1</td>
<td>Vectors</td>
<td>9</td>
</tr>
<tr>
<td>2.2</td>
<td>Matrices</td>
<td>10</td>
</tr>
<tr>
<td>2.3</td>
<td>Matrix Diagonalization</td>
<td>11</td>
</tr>
<tr>
<td>2.4</td>
<td>Some Important $2 \times 2$ matrices</td>
<td>12</td>
</tr>
<tr>
<td>2.5</td>
<td>Properties of Matrices</td>
<td>13</td>
</tr>
<tr>
<td><strong>3</strong></td>
<td>Introduction to Quantum Mechanics</td>
<td>15</td>
</tr>
<tr>
<td>3.1</td>
<td>Quantum States as Complex Vectors</td>
<td>15</td>
</tr>
<tr>
<td>3.2</td>
<td>Phases</td>
<td>17</td>
</tr>
<tr>
<td>3.3</td>
<td>Observables</td>
<td>18</td>
</tr>
<tr>
<td>3.4</td>
<td>Outer Product Notation</td>
<td>21</td>
</tr>
<tr>
<td>3.5</td>
<td>Functions of operators</td>
<td>22</td>
</tr>
<tr>
<td>3.6</td>
<td>Measurements</td>
<td>23</td>
</tr>
<tr>
<td>3.7</td>
<td>Statistics of Measurements</td>
<td>25</td>
</tr>
<tr>
<td>3.8</td>
<td>Composite Systems</td>
<td>26</td>
</tr>
<tr>
<td>3.9</td>
<td>Generalized Born Rule</td>
<td>27</td>
</tr>
<tr>
<td>3.10</td>
<td>The Uncertainty Principle</td>
<td>28</td>
</tr>
<tr>
<td>3.11</td>
<td>Time Evolution of Quantum States</td>
<td>29</td>
</tr>
<tr>
<td><strong>4</strong></td>
<td>General state of a qubit, no-cloning theorem, entanglement and Bell states</td>
<td>31</td>
</tr>
<tr>
<td>4.1</td>
<td>General qubit states</td>
<td>31</td>
</tr>
<tr>
<td>4.2</td>
<td>No cloning theorem</td>
<td>33</td>
</tr>
<tr>
<td>4.3</td>
<td>Entanglement and Bell states</td>
<td>34</td>
</tr>
<tr>
<td>4.4</td>
<td>Angular Momentum Eigenstates</td>
<td>35</td>
</tr>
<tr>
<td><strong>5</strong></td>
<td>The Density Matrix</td>
<td>39</td>
</tr>
<tr>
<td>5.1</td>
<td>Introduction</td>
<td>39</td>
</tr>
<tr>
<td>5.2</td>
<td>Definition of the Density Matrix</td>
<td>39</td>
</tr>
<tr>
<td>5.3</td>
<td>Determining if a state is entangled</td>
<td>41</td>
</tr>
<tr>
<td>5.4</td>
<td>Some Simple Examples</td>
<td>42</td>
</tr>
</tbody>
</table>
## CONTENTS

<table>
<thead>
<tr>
<th>Section</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>5.4.1 Example 1</td>
<td>42</td>
</tr>
<tr>
<td>5.4.2 Example 2</td>
<td>43</td>
</tr>
<tr>
<td>5.4.3 Example 3</td>
<td>44</td>
</tr>
<tr>
<td>5.5 Conclusions</td>
<td>45</td>
</tr>
<tr>
<td>5.A Schmidt Decomposition</td>
<td>46</td>
</tr>
<tr>
<td>5.B Change in the density matrix under a unitary transformation</td>
<td>46</td>
</tr>
<tr>
<td>6 Coherent Superposition Versus Incoherent Addition of Probabilities</td>
<td>49</td>
</tr>
<tr>
<td>6.1 Coherent Linear Superposition: 1 qubit</td>
<td>49</td>
</tr>
<tr>
<td>6.2 Incoherent (Classical) Addition of Probabilities: 2 qubits</td>
<td>50</td>
</tr>
<tr>
<td>6.A Computation of the Density Matrix $\rho^A$ after the action of the Hadamard</td>
<td>51</td>
</tr>
<tr>
<td>7 Einstein-Podolsky-Rosen (EPR), Bell's inequalities, and Local Realism</td>
<td>53</td>
</tr>
<tr>
<td>7.1 Introduction</td>
<td>53</td>
</tr>
<tr>
<td>7.2 An EPR Experiment</td>
<td>54</td>
</tr>
<tr>
<td>7.3 Bells' Inequality</td>
<td>55</td>
</tr>
<tr>
<td>7.A The spin-singlet state is isotropic</td>
<td>59</td>
</tr>
<tr>
<td>8 Classical and Quantum Gates</td>
<td>61</td>
</tr>
<tr>
<td>8.1 Classical Gates</td>
<td>61</td>
</tr>
<tr>
<td>8.2 Quantum Gates</td>
<td>63</td>
</tr>
<tr>
<td>9 Generating and measuring Bell States</td>
<td>69</td>
</tr>
<tr>
<td>10 Quantum Functions</td>
<td>71</td>
</tr>
<tr>
<td>10.1 An elementary quantum function</td>
<td>71</td>
</tr>
<tr>
<td>10.2 Quantum Parallelism</td>
<td>72</td>
</tr>
<tr>
<td>11 Deutsch's Algorithm</td>
<td>75</td>
</tr>
<tr>
<td>11.1 Introduction</td>
<td>75</td>
</tr>
<tr>
<td>11.A An alternative derivation</td>
<td>78</td>
</tr>
<tr>
<td>11.B Derivation of some useful identities in quantum circuits</td>
<td>79</td>
</tr>
<tr>
<td>12 The Bernstein-Vazirani Algorithm</td>
<td>83</td>
</tr>
<tr>
<td>12.1 The Algorithm</td>
<td>83</td>
</tr>
<tr>
<td>12.2 An Alternative Derivation</td>
<td>85</td>
</tr>
<tr>
<td>12.A Homework Problem on the Deutsch-Josza Algorithm</td>
<td>86</td>
</tr>
<tr>
<td>12.B Homework problem on making a Toffoli Gate out of 1-qubit and 2-qubit gates</td>
<td>88</td>
</tr>
<tr>
<td>13 Simon's Algorithm</td>
<td>89</td>
</tr>
<tr>
<td>14 Factoring and RSA (Rivest-Shamir-Adleman) Encryption</td>
<td>93</td>
</tr>
<tr>
<td>14.A The Euclidean Algorithm</td>
<td>95</td>
</tr>
<tr>
<td>14.B Extension of the Euclidean Algorithm to find an inverse modulo an integer</td>
<td>96</td>
</tr>
<tr>
<td>15 Using Period Finding to Factor an Integer</td>
<td>97</td>
</tr>
</tbody>
</table>
Preface

This material has been given as a one-quarter course for undergraduates in the physical sciences at the University of California Santa Cruz. When supplemented by homework problems it could provide the basis for an undergraduate course at other institutions as well.

In order that the course be accessible to majors other than physics, the rules of quantum mechanics were taught from scratch in the first part. While some of my physics colleagues were surprised that this could be done, it is perfectly feasible because much of what is included in a traditional physics course on quantum mechanics concerns continuous degrees of freedom so one has to cover complicated topics such as partial differential equations, boundary conditions, angular momentum, and a plethora of special functions. All this can be omitted in a quantum computing course which is focused on 2-state systems. A solid background in linear algebra is required. A brief review of this is given at the start, but the treatment is fast and it is assumed that the students will have seen the material before.

The aim of the course is to get students to the level where they can understand the two most important topics covered: Shor’s algorithm in Chapter 18 and quantum error correction in Chapter 19. Unlike quantum algorithms proposed previously, Shor’s algorithm for factoring integers gives a spectacular speedup on a problem of practical importance (encryption of data sent down a public channel). Considerable experimental challenges remain to implement Shor’s algorithm for a large number of qubits but quantum error correction will be essential in order to achieve this, because qubits are highly susceptible to noise. Incorporating quantum error correction still leaves huge experimental challenges before achieving the goal of factoring integers larger than what is possible classically, but without quantum error correction it would clearly be impossible.

The goal, then, is to present a course at the undergraduate level, but which still goes into enough depth to give a good understanding of Shor’s algorithm and the basics of quantum error correction. No details will be given on the many experimental approaches to building a quantum computer, which is a huge topic that would merit a separate course in its own right.

There are, of course, excellent more advanced texts, such as the monumental classic by Nielsen and Chuang [NC00] and the books by Mermin [Mer07] and Rieffel and Pollack [RP14]. The book closest in level and spirit to the present text is the one by Vathsam [Vat16], which I found very useful when preparing this material. My hope is that this these lectures will take students to a level where they can follow the rapidly-moving advanced literature in the field.

Peter Young
University of California, Santa Cruz
August 3, 2020
Chapter 1

The Strange World of Quantum Mechanics

1.1 Introduction

The quantum world is strange, and different from the classical world that we see around us. Our intuition obtained from everyday experience is for objects that we can see. It does not apply to the quantum world where we are dealing with very small objects, objects that (in most cases) are too small to see. I give two quotations, from eminent physicists, which illustrate the strangeness of the quantum world:

“Anyone who is not shocked by quantum mechanics hasn’t understood it.”
(Attributed to Niels Bohr).

“I think I can say that nobody understands quantum mechanics.”
(Richard Feynman).

More information on the topic of this chapter can be found in Refs. [NC00, Mer07, Vat16] and in Ch. 1, Vol. 3 of the Feynman Lectures on Physics [FLS64].

The main question to be addressed in this course is whether we can use the difference between the quantum and classical worlds to find more efficient algorithms to solve certain problems by treating the data in a quantum computer in which it is processed according to quantum rules rather than classical rules. We shall see that for some problems the answer is “yes”. I should mention now that there is a practical question of whether we can actually build a useful quantum computer. The difficulties of building such a device have not yet been overcome, though much progress has been made. In this course, which focuses on theory, we will not describe in detail the many experimental approaches that are being implemented to try to achieve this goal. However, we will discuss in Chapter 19 how one can reduce errors caused by an imperfect device, a topic called “Quantum Error Correction”.

A quantum computer, then, is one in the data is processed by quantum, rather than classical rules. What do we mean by this? In a classical computer the data is stored in bits, which take two values 0 and 1. A quantum computer also uses 2-state systems called qubits. We indicate these two states by $|0\rangle$ and $|1\rangle$, a notation introduced by the physicist Paul Dirac. The difference from classical bits is that the general state of a qubit, which we will write as $|\psi\rangle$, is a superposition of states $|0\rangle$ and $|1\rangle$:

$$|\psi\rangle = \alpha|0\rangle + \beta|1\rangle,$$

(1.1)

where $\alpha$ and $\beta$ are numbers (complex in general). For reasons that will be explained later, we need the condition $|\alpha|^2 + |\beta|^2 = 1$. One sometimes says loosely that a qubit in the state described by Eq. (1.1)
is simultaneously in states $|0\rangle$ and $|1\rangle$. This is to be contrasted with a classical bit which takes value 0 or 1.

Our main goal in this course will be to see if one can gain computationally from superposition states.

Next we will discuss some experiments which illustrate that quantum mechanics is strange.

### 1.2 The Two-Slit Experiment

You are probably familiar with experiments involving light going through slits which demonstrate that light, being a wave, shows interference.

First consider just one slit. If the slit width $d$ is very large compared with the wavelength of light $\lambda$ (the geometrical optics limit) then, to a good approximation, the light continues in a straight line. However, if the slit width is comparable to, or less than, $\lambda$, the light spreads out after passing through the slit, which is called diffraction. Figure 1.1 sketches the intensity of light observed on a screen behind the slit.

![Figure 1.1: A beam of light spreads out (diffractions) when passing though a slit of width $d$ which is comparable to, or smaller than, the wavelength of the light $\lambda$. The figure shows a sketch of the intensity of the beam on a screen after it has passed through the slit.](image)

Next we will discuss some experiments which illustrate that quantum mechanics is strange.

![Figure 1.2: A two slit experiment. Interference fringes, oscillations of strong and weak intensity, are seen due destructive and constructive interference. The overall envelope of the intensity has a similar form to that from a single slit shown in Fig. 1.1](image)
If the light beam passes two slits, as shown in Fig. 1.2, one observes interference fringes, oscillations of strong and weak intensity, due to interference between between the beams going through the two slits. If the difference in path length \( |r_1 - r_2| \) (see Fig. 1.3) satisfies \( |r_1 - r_2| = n\lambda \) (for integer \( n \)) one has constructive interference and a maximum intensity, whereas if \( |r_1 - r_2| = (n + \frac{1}{2})\lambda \) one has a minimum intensity. Hence, as one moves along the screen one alternately gets regions of low intensity and high intensity. These are called interference fringes.

This is the classical picture. That is, we shine a beam at the two slits, some of it goes through one slit, some goes through the other slit, and when these two beams recombine they interfere.

![Figure 1.3: The difference in the length of the paths taken by the beams going through the two slits is \( |r_1 - r_2| \). This varies as a function of the location on the screen, so the interference changes from constructive, where \( |r_1 - r_2| = n\lambda \), to destructive, where \( |r_1 - r_2| = (n + \frac{1}{2})\lambda \) with \( n \) an integer.](image)

Now we reduce the intensity of the light. At some point we notice that light is not a continuous wave but consists of discrete bunches of energy called photons. To detect individual photons, we place an array of photon counters on the screen and count the number of discrete clicks in each counter, see Fig. 1.4. We ask how the number of clicks varies for the counters at different points on the screen.

![Figure 1.4: The two slit experiment where individual photons are detected by an array of counters which count the number of photons at their location.](image)

Suppose we reduce the intensity so much that the time between emitting photons is greater than the time it takes a photon to pass through the experimental setup, i.e. the photons go through one at a time. Do we see an interference pattern? Using our classical intuition we would say “no” because surely each photon “must” either goes through the upper slit or the lower one and can therefore not interfere with itself. In other words we would expect the intensity of clicks in the counters to vary smoothly along the screen, as in the classical single slit experiment shown in Fig. 1.1.
CHAPTER 1. THE STRANGE WORLD OF QUANTUM MECHANICS

Amazingly, this is not so and we do see an interference pattern. In other words the number of clicks in the counters varies rapidly and in an oscillatory manner as we move along the screen, just as in the classical two-slit experiment shown in Fig. 1.2. It looks as though a single photon does go through both slits. You may already be feeling (correctly) that this looks suspiciously like a superposition state such as the one we wrote down in Eq. (1.1), where now \(|0\rangle\) refers to photon through the upper slit and \(|1\rangle\) to photon through the lower slit.

You might ask “why don’t we just look and see which slit the photon went through”. Well, photons being electrically neutral are hard to observe unless we absorb them (which we want to do only when they reach the screen). Scattering of one photon by another is immeasurably small. So, with photons we can’t observe which slit they went through. However, we can do the same experiment with electrons rather than photons. Like photons, electrons have both particle and wave-like properties, but, being charged, they readily scatter light so we can see observe them by shining light on them. The discussion which follows is based on Ch. 1, Vol. 3 of Feynman [FLS64].

In this new version of the experiment we send electrons through the slits one at a time. To see which slit they went through we shine light of wavelength \(\lambda\) at the slits and observe a flash of light every time an electron goes through.

Suppose that we choose a light source that has a wavelength \(\lambda\) which is bigger than the slit spacing \(d\). We do see a flash every time an electron passes through, and observe that there is still an interference pattern but, the flash of light is of size \(\lambda\) which is greater than the separation of the slits, so we can’t tell which slit the electron went through. Clearly we need to use a light source with wavelength less than \(d\). When we do this, indeed we see a flash at either the upper slit or the lower slit every time an electron passes, so we’ve achieved our goal of observing which slit each electron goes through. But alas, when we look at the counts registered on the detectors we see that the interference fringes have been washed out, and we have just a smooth variation in the number of clicks along the screen. Observations such as these show that it is not possible to determine which slit each electron goes through and observe interference fringes.

This observation guides us to a second piece of intuition regarding quantum mechanics (the first, mentioned above, is that a quantum system can be in a superposition state). We see that a measurement can unavoidably change a quantum state, and in particular can destroy a superposition.

Classically, measurements are passive, and can be done in a delicate way so they simply reveal a reality which is already present whether we observe it or not. Quantum mechanically, measurements play a much more active role and can change the state of the system. In particular, we shall see that if we observe a system in a particular state, we can’t necessarily say that it was in that state before the measurement.

1.3 Stern-Gerlach Experiment

We will now discuss a second experiment which gives additional insight into superposition states.

Consider the hydrogen atom, which has one proton, the nucleus, and one electron. In its ground state the electron has a symmetric distribution of velocities and so there is no net circulating current around the proton. Hence the electron orbital motion does not give rise to a magnetic moment which could interact with an external magnetic field. However, the electron has an internal state, called spin, which gives rise to a magnetic moment \(\vec{\mu}\), proportional to the spin angular momentum.

In a magnetic field, which we will take to be in the \(z\)-direction, there is a force on a magnetic moment if the field is non-uniform. To see this, recall that the energy of a magnetic moment in a

---

1The proton also has a spin and hence a magnetic moment but, because of its much larger mass, its magnetic moment is much smaller than that of the electron and so does not play a role in our discussion.
magnetic field $\vec{B}$ is $-\vec{\mu} \cdot \vec{B}$ and therefore the force is given by

$$\vec{F} = \vec{\nabla} (\vec{\mu} \cdot \vec{B}) = \vec{\nabla} (\mu_z B_z (z))$$

(1.2)

and hence

$$F_z = \mu_z \frac{dB_z}{dz},$$

(1.3)

where we have assumed, without loss of generality, that the field changes as function of $z$. Hence a beam of hydrogen atoms in a non-uniform field varying in the $z$-direction will be deflected by an amount proportional to the $z$-component of $\vec{\mu}$.

We send in a beam of unpolarized hydrogen atoms into a non-uniform field. This is the famous Stern-Gerlach (SG) experiment. Since the direction of $\vec{\mu}$ is random, classically $\mu_z$ takes a range of values, so we would expect a continuous range of deflections. However, it is found that only two beams emerge, which are deflected in opposite directions, see Fig. 1.5. It seems that the spin component along $z$ has only two components, corresponding to states which we might label as $| \uparrow_z \rangle$ and $| \downarrow_z \rangle$.

Figure 1.5: The Stern-Gerlach apparatus.

Now suppose that we orientate the magnet so its field is in the $x$-direction. Again we will see two beams emerging, indicating that $\mu_x$ has only two possible values $| \uparrow_x \rangle$ and $| \downarrow_x \rangle$.

How are $| \uparrow_z \rangle$ and $| \downarrow_z \rangle$ related to $| \uparrow_x \rangle$ and $| \downarrow_x \rangle$? We can get an idea of this if we run our beam first through a SG setup with the field in the $z$-direction and then pass one of the resulting beams through an SG setup in the $x$-direction as shown in Fig. 1.6. The final result is found to be two beams of equal intensity.

It looks as though $| \uparrow_z \rangle$ can be thought of as $| \uparrow_x \rangle$ with probability $1/2$ and $| \downarrow_x \rangle$ with probability $1/2$. We will see in a future lecture that $| \uparrow_z \rangle$ is actually a superposition of $| \uparrow_x \rangle$ and $| \downarrow_x \rangle$ as follows:

$$| \uparrow_z \rangle = \frac{1}{\sqrt{2}} (| \uparrow_x \rangle + | \downarrow_x \rangle),$$

(1.4)

\footnote{The electron is the simplest two-state system so, in principle, it could be used as a qubit, but because it is charged it interacts too strongly with its environment to be useful.}
where we say that there is an amplitude \( 1/\sqrt{2} \) for \(| \uparrow_z \rangle\) to be \(| \uparrow_x \rangle\) and amplitude \( 1/\sqrt{2} \) for it to be \(| \downarrow_x \rangle\). As we shall also see later, the probability that a measurement gives a certain result is the square of the corresponding amplitude\(^3\) so the probability of measuring \(| \uparrow_x \rangle\) after the \(SG_x\) apparatus is \(1/2\) (as observed) and the same for \(| \downarrow_x \rangle\).

It is also true that
\[
| \uparrow_x \rangle = \frac{1}{\sqrt{2}} (| \uparrow_z \rangle + | \downarrow_z \rangle),
\]
so if we run one of the beams from the \(SG_x\) apparatus in Fig. 1.6 through another \(SG_x\) apparatus we will get beams with equal intensity for \(| \uparrow_z \rangle\) and \(| \downarrow_z \rangle\), see Fig. 1.7. Note a surprising aspect of this result. After the first \(SG_z\) apparatus, there is zero probability for getting \(| \uparrow_z \rangle\), but after the \(SG_x\) apparatus there is a 50% probability for finding \(| \uparrow_z \rangle\). This is a clear example of a measurement (in this case that done by the \(SG_x\) apparatus) affecting the state of the system.

---

### 1.4 Photons

In the previous section we noted that the spin of the electron is a two-state quantum system. Here we discuss another two-state quantum system, the photon, the quantum of light.

\(^3\)The fact that probabilities add to 1, is why \(|\alpha|^2 + |\beta|^2 = 1\) in Eq. (1.1).
Light is a transverse electric field, in which the electric field $\vec{E}$ and magnetic field $\vec{B}$ are perpendicular both to each other and to the direction of propagation specified by the wavevector $\vec{k}$. For example, if $\vec{E}$ is in the $x$ direction, $\vec{B}$ in the $y$ direction, and $\vec{k}$ in the $z$ direction we have

\[ \vec{E} = E_0 \hat{x} e^{i(kz-\omega t)}, \]
\[ \vec{B} = B_0 \hat{y} e^{i(kz-\omega t)}. \] (1.6)

The direction of $\vec{E}$ is called the polarization direction. There are two distinct polarizations which we can call “vertical”,

\[ |\downarrow\rangle, \quad \text{equivalent to} \quad |^\downarrow_z\rangle \equiv |0\rangle, \] (1.7)

and “horizontal”

\[ |\leftrightarrow\rangle, \quad \text{equivalent to} \quad |^\leftrightarrow_z\rangle \equiv |1\rangle. \] (1.8)

What are the analogs of $|\uparrow_x\rangle$ and $|\downarrow_x\rangle$? The answer is diagonal polarizations:

\[ |\uparrow\rangle \equiv \frac{1}{\sqrt{2}} (|\downarrow\rangle + |\leftrightarrow\rangle), \quad \text{equivalent to} \quad |\uparrow_x\rangle \equiv \frac{1}{\sqrt{2}} (|0\rangle + |1\rangle), \] (1.9)
\[ |\downarrow\rangle \equiv \frac{1}{\sqrt{2}} (|\downarrow\rangle - |\leftrightarrow\rangle), \quad \text{equivalent to} \quad |\downarrow_x\rangle \equiv \frac{1}{\sqrt{2}} (|0\rangle - |1\rangle). \]

Photons do not interact with each other to a measurable extent, and can not readily be stored, but have the advantage that they can be transmitted over great distances down optical fibers, preserving their polarization. These properties will be useful for some quantum protocols to be discussed in Chapter 21.
Chapter 2

Review of Linear Algebra

The theory of quantum mechanics is based on linear algebra which is a pre-requisite for the course and is standard material available in many books. In this chapter we summarize those topics in linear algebra which students in the class are expected to be already familiar with.

2.1 Vectors

An abstract vector $\vec{v}$ can be represented in terms of its $N$ components $v_i, (i = 1, \cdots, N)$

$$\vec{v} = \sum_{i=1}^{N} v_i \hat{e}_i,$$  (2.1)

with respect to a set of basis vectors $\hat{e}_i$, which form an orthonormal set, i.e.

$$\vec{e}_i \cdot \vec{e}_j = \delta_{ij},$$  (2.2)

where the left hand side is a scalar product

$$\vec{a} \cdot \vec{b} = \sum_{i=1}^{N} a_i b_i,$$  (2.3)

and $\delta_{ij}$ is the Kronecker delta function,

$$\delta_{ij} = \begin{cases} 1 & (i = j), \\ 0 & (i \neq j), \end{cases}$$  (2.4)

The number of independent basis states required to represent any vector is called the size of the “vector space”. It is denoted here by $N$.

The vector $\vec{v}$ can be represented as a column vector

$$\vec{v} = \begin{pmatrix} v_1 \\ v_2 \\ \vdots \\ v_N \end{pmatrix},$$  (2.5)

and its transpose as a row vector

$$\vec{v}^T = (v_1 \ v_2 \ \cdots \ v_N).$$  (2.6)
The length of a vector is given by

\[ |v| = \left( \sum_{i=1}^{N} v_i^2 \right)^{1/2} = (\vec{v} \cdot \vec{v})^{1/2}. \] (2.7)

One can represent a vector with respect to different orthonormal bases rotated with respect to each other. If a vector has components \( v'_i \) with respect to the new basis, there is a linear relation between the old and new components,

\[ \vec{v}' = M \vec{v}, \] (2.8)

or, in terms of components

\[ v'_i = \sum_{j=1}^{N} M_{ij} v_j, \] (2.9)

where \( M \) is an \( N \times N \) matrix with elements \( M_{ij} \). In order that \( M \) describes a rotation (which preserves lengths of vectors and angles between them), it is necessary that \( M \) be an orthogonal matrix, i.e.

\[ M^{-1} = M^T, \] (2.10)

where \( M^T \) is the transpose matrix, and \( M^{-1} \) is the matrix inverse which means that \( M^{-1}M = MM^{-1} = 1 \) where \( 1 \) is the identity matrix. An example of a rotation matrix for two-component vectors is

\[ M = \begin{pmatrix} \cos \theta & \sin \theta \\ -\sin \theta & \cos \theta \end{pmatrix}, \] (2.11)

where \( \theta \) is the rotation angle.

The scalar product of two vector products is independent of basis, so

\[ \vec{a} \cdot \vec{b} = \sum_{i=1}^{N} a_i b_i = \sum_{i=1}^{N} a'_i b'_i. \] (2.12)

This is why \( \vec{a} \cdot \vec{b} \) is called the scalar product.

If \( \vec{a} \cdot \vec{b} = 0 \) we say that the vectors \( a \) and \( b \) are orthogonal.

### 2.2 Matrices

Matrix multiplication. If \( A \) and \( B \) are matrices then the matrix product \( C = AB \) is given in terms of its elements by

\[ C_{ij} = \sum_{k=1}^{m} A_{ik} B_{kj}. \] (2.13)

If \( A \) is of dimension \( N \times m \) (\( N \) rows and \( m \) columns) then \( B \) must have \( m \) rows. If \( B \) has \( p \) columns then \( C \) is of dimension \( N \times p \). It will sometimes be useful to think of a column vector as an \( N \times 1 \) dimensional matrix (\( N \) rows and 1 column), and a row vector as a \( 1 \times N \) dimensional matrix. Apart from vectors, the matrices in this course will be square (number of rows equals number of columns).

Matrix multiplication has the property that the order of multiplication matters in general. If \( AB \neq BA \) we say that the matrices don’t commute. Lack of commutation of matrices will have important consequences in quantum mechanics.

Some special types of matrices:

- Symmetric: \( M^T = M \) (\( M^T \) is the transpose, so \( (M^T)_{ij} = M_{ji} \)).
2.3. MATRIX DiAGONALIZATION

- Orthogonal: $M^T = M^{-1}$ ($M^{-1}$ is the inverse, so $M^{-1}M = 1$, the identity matrix).

When dealing with complex matrices, one is usually interested in Hermitian matrices rather than symmetric ones, and unitary matrices rather than orthogonal ones, where these are defined by:

- Hermitian: $M^\dagger = M$ ($M^\dagger$ is the adjoint, the complex conjugate of the transpose ($M^T$)*).

- Unitary: $M^\dagger = M^{-1}$.

Hermitian and unitary matrices play important roles in quantum mechanics.

In quantum mechanics, we need complex vectors as well as complex matrices. The main new feature compared with real vectors is a slight difference in the definition of the scalar product, namely one takes the complex conjugate of the left hand vector, i.e.

\[ \bar{a} \cdot \bar{b} = \sum_{i=1}^{N} a_i^* b_i. \]  

(2.14)

In terms of rules for matrix multiplication one can view the scalar product as the matrix product of the complex conjugate of the transpose vector (row vector) for $a$ with the vector (column vector) for $b$, i.e.

\[ \bar{a} \cdot \bar{b} \equiv (a^T)^* b = (a_1^* \ a_2^* \ \cdots \ a_N^*) \begin{pmatrix} b_1 \\ b_2 \\ \vdots \\ b_N \end{pmatrix}, \]  

(2.15)

in which $(a^T)^*$ is an $1 \times N$ dimensional matrix, $b$ is an $N \times 1$ dimensional matrix, and $(a^T)^* b$ denotes matrix multiplication with the result being a single number (scalar).

The length of a complex vector is still the square root of the scalar product of the vector with itself, i.e.

\[ |a| = (\bar{a} \cdot \bar{a})^{1/2} = \left(\sum_{i=1}^{N} |a_i|^2\right)^{1/2}. \]  

(2.16)

### 2.3 Matrix Diagonalization

Let $A$ be $N \times N$ matrix and $\bar{x}$ an $N$-component vector. Then if $A\bar{x}$ is proportional to $\bar{x}$ itself, i.e. if

\[ A\bar{x} = \lambda \bar{x} \]  

or, in terms of elements,

\[ \sum_{j=1}^{N} A_{ij} x_j = \lambda x_i, \]  

(2.17)

then we say that $\lambda$ is an eigenvalue and $\bar{x}$ the corresponding eigenvector of $A$. There are $N$ eigenvalues which may not all be distinct. If two or more eigenvalues are equal we say that they are degenerate. We can always multiply an eigenvector by a constant and it remains an eigenvector. In quantum mechanics we will need to choose this multiplicative constant so the vector is “normalized”, i.e. has unit length. Two vectors which are normalized and orthogonal are said to be “orthonormal”.

The eigenvalues are obtained from solving

\[ \det(A - \lambda I) = 0, \]  

(2.18)

where $\det$ is short for determinant. Expanding out the determinant gives an $N$-th order polynomial equation for $\lambda$.

The eigenvalues and eigenvectors of Hermitian matrices have special properties:
• The eigenvalues are all real.

• Eigenvectors corresponding to unequal (non-degenerate) eigenvalues are orthogonal. For eigenvectors corresponding to degenerate eigenvalues, one can form linear combinations which are orthogonal.

We also need to know a useful property about unitary matrices, namely that the rows form orthonormal vectors, as do the columns. To determine if a matrix is unitary it may be easier to do this check rather than compute the inverse.

If we consider two \( N \times N \) matrices \( A \) and \( B \), one can show that they have the same eigenvectors if and only if the matrices commute, i.e. if \([A, B] \equiv AB - BA = 0\). This result will have important consequences in quantum mechanics.

### 2.4 Some Important 2 \( \times \) 2 matrices

Important examples of Hermitian matrices are the Pauli (spin) matrices

\[
X = \begin{pmatrix} 0 & 1 \\ 1 & 0 \end{pmatrix}, \quad Y = \begin{pmatrix} 0 & -i \\ i & 0 \end{pmatrix}, \quad Z = \begin{pmatrix} 1 & 0 \\ 0 & -1 \end{pmatrix},
\]

(called \( \sigma_x, \sigma_y \) and \( \sigma_z \) in the physics literature).

We will now show that any 2 \( \times \) 2 matrix can be expressed as a linear combination of the three Pauli matrices plus the identity. To see this note that \( X, Y, Z \) and \( \mathbb{1} \) are linearly independent (we can’t write one as a linear combination of the others). Also a general 2 \( \times \) 2 matrix

\[
A = \begin{pmatrix} t & u \\ v & w \end{pmatrix}
\]

has 4 complex elements, and so a total of 8 real parameters. If we write

\[
A = a_0 \mathbb{1} + a_x X + a_y Y + a_z Z
\]

then there are also 4 complex coefficients (8 real parameters). Hence there are just the right number of coefficients to specify any 2 \( \times \) 2 matrix, so Eq. (2.21) is a general expression for a 2 \( \times \) 2 matrix.

Let's determine the eigenvalues and eigenvectors of \( X \). The eigenvalues \( \lambda \) are obtained from

\[
\begin{vmatrix} 0 - \lambda & 1 \\ 1 & 0 - \lambda \end{vmatrix} = 0,
\]

which gives \( \lambda^2 - 1 = 0 \) or \( \lambda = \pm 1 \). These are real, which they must be since \( X \) is Hermitian.

Let us now get the eigenvectors. We denote the corresponding normalized eigenvectors by \( \vec{e}_{+1} \) and \( \vec{e}_{-1} \) with coefficients \( a \) and \( b \).

- \( \lambda = +1 \).

\[
\begin{pmatrix} 0 & 1 \\ 1 & 0 \end{pmatrix} \begin{pmatrix} a \\ b \end{pmatrix} = \begin{pmatrix} a \\ b \end{pmatrix},
\]

which gives the equations \( b = a \) and \( a = b \) (these are equivalent). To normalize the eigenvector, we take \( a = b = 1/\sqrt{2} \), so

\[
\vec{e}_{+1} = \frac{1}{\sqrt{2}} \begin{pmatrix} 1 \\ 1 \end{pmatrix}.
\]
• $\lambda = -1$.

$$
\begin{pmatrix}
0 & 1 \\
1 & 0
\end{pmatrix}
\begin{pmatrix}
a \\
b
\end{pmatrix} =
\begin{pmatrix}
a \\
b
\end{pmatrix}
$$  
(2.25)

which gives the two equations $b = -a$ and $a = -b$ (which are equivalent). The normalized eigenvector is therefore

$$
\vec{e}_{-1} = \frac{1}{\sqrt{2}} \begin{pmatrix}
1 \\
-1
\end{pmatrix}.
$$  
(2.26)

The eigenvectors $\vec{e}_{+1}$ and $\vec{e}_{-1}$ are orthogonal, as we know they must be since $X$ is Hermitian.

It is instructive for the student to show that the eigenvalues of $Y$ and $Z$ are also $\pm 1$ and to determine their eigenvectors.

The student should also be able to show that $X, Y$ and $Z$ are not only Hermitian but also unitary. Another $2 \times 2$ matrix which is very important in quantum computing is the Hadamard, defined by

$$
H = \frac{1}{\sqrt{2}} (X + Z) = \frac{1}{\sqrt{2}} \begin{pmatrix}
1 & 1 \\
1 & -1
\end{pmatrix}.
$$  
(2.27)

2.5 Properties of Matrices

Two properties of square matrices will be important: the trace, which is the sum of the diagonal elements, and the determinant. It is left as an exercise for the student to show (i) that the trace is the sum of the eigenvalues, and (ii) that the trace of a product of matrices is invariant under a cyclic permutation of the matrices so, for example, $\text{Tr} AB = \text{Tr} BA$ even if $A$ and $B$ don’t commute so $AB \neq BA$.

We will now show that the determinant is the product of the eigenvalues. We note first that an $N \times N$ matrix $A$ can be diagonalized as follows:

$$
D = S^{-1} AS,
$$  
(2.28)

where $D$ is a diagonal matrix with the eigenvalues of $A$ on the diagonal,

$$
D = \begin{pmatrix}
\lambda_1 & 0 & \cdots & 0 \\
0 & \lambda_2 & \cdots & 0 \\
\vdots & \vdots & \ddots & \vdots \\
0 & 0 & \cdots & \lambda_N
\end{pmatrix},
$$  
(2.29)

and the matrix $S$, which effects the diagonalization, is constructed out of the eigenvectors of $A$. Multiplying Eq. (2.28) on the left by $S$ and on the right by $S^{-1}$ gives

$$
A = SDS^{-1}.
$$  
(2.30)

An important but not very well known result of linear algebra is that determinant of a product of matrices is equal to the product of the determinants, i.e.

$$
\det (AB) = \det A \det B.
$$  
(2.31)

\footnote{There are some matrices with degenerate eigenvalues which have less than $N$ independent eigenvectors. These can not be diagonalized. However, this situation does occur for Hermitian or unitary matrices, the two categories that are of principle interest in quantum mechanics, and so we will ignore non-diagonalizable matrices in this course.}
Taking the determinant of both sides of Eq. (2.30) gives

\[
\det A = \det S \det D \det S^{-1} \\
= \det D \det S \det S^{-1} \\
= \det D \det (SS^{-1}) \\
= \det D = \prod_{m=1}^{N} \lambda_m, \quad (2.32)
\]

which is the desired result.
Chapter 3

Introduction to Quantum Mechanics

In this chapter we give an introduction to quantum mechanics. A good textbook on the subject, at an undergraduate level, is Griffiths [Gri05].

3.1 Quantum States as Complex Vectors

In Chapter 2 we reviewed linear algebra, including vectors, generalized to the case where the coefficients of the vectors are complex.

We now describe the basic postulates of quantum mechanics. We will see that the framework is precisely that of complex vectors. The notation, however, is quite different and so, for the next few equations, we will show both a statement concerning quantum mechanics in quantum mechanics notation, and the corresponding statement for complex vectors in the standard notation of linear algebra.

While the discussion which follows may seem very abstract don’t forget that quantum mechanics is arguably the most successful theory in all of physics, with countless precise comparisons between theory and experiment, some to the most exquisite accuracy\footnote{For example, experimental and theoretical values for the magnetic moment of the electron agree to better than a part in a trillion, see \url{https://www.mdpi.com/2218-2004/7/2/45/pdf}.}

Now we get started with quantum mechanics:

\textbf{Ansatz 1:} The state of a quantum system is a complex vector (which we shall often call a “state vector” or just a vector).

In quantum computing one uses the notation of Dirac, in which a quantum state is written as $|\psi\rangle$.

\begin{equation}
\text{QM state : } |\psi\rangle, \quad \iff \quad \text{complex vector : } \vec{v}.
\end{equation}

(3.1)

In equations with the double arrow $\iff$ in the middle, the part to the left of the arrow is in the notation of quantum mechanics, and the part to the right is the corresponding statement in standard vector notation. The state $|\psi\rangle$ can be expressed as a linear combination of basis states $|n\rangle$,

\begin{equation}
|\psi\rangle = \sum_{n=1}^{N} c_n |n\rangle, \quad \iff \quad \vec{v} = \sum_{n=1}^{N} v_n \hat{e}_n.
\end{equation}

(3.2)

We can write the state as a column vector

\begin{equation}
|\psi\rangle = \begin{pmatrix} c_1 \\ c_2 \\ \vdots \\ c_N \end{pmatrix}, \quad \iff \quad \vec{v} = \begin{pmatrix} v_1 \\ v_2 \\ \vdots \\ v_N \end{pmatrix}.
\end{equation}

(3.3)
We also introduce the dual state vector, denoted by $\langle \psi |$. This corresponds to the complex conjugate of the transpose vector introduced in Eq. (2.15) in the context of the scalar product of a complex vector. If $|\phi \rangle$ is represented as a column vector by

$$
|\phi \rangle = \begin{pmatrix} d_1 \\ d_2 \\ \vdots \\ d_N \end{pmatrix},
$$

(3.4)

then the corresponding dual vector is

$$
\langle \phi | = (d^*_1 \: d^*_2 \cdots \: d^*_N) \quad \iff \quad (a^T)^* = (a^*_1 \: a^*_2 \cdots \: a^*_N),
$$

(3.5)
i.e. a row vector in which the coefficients are the complex conjugate of the original column vector.

The scalar product of two vectors is called the “inner product” in a general context and this nomenclature will be used here from now on. In quantum mechanics, the inner product of a vector $|\psi \rangle$ with vector $|\phi \rangle$ is written as $\langle \phi | \psi \rangle$.

$$
\langle \phi | \psi \rangle = \sum_{n=1}^{N} d^*_n c_n \quad \iff \quad \vec{a} \cdot \vec{b} = \sum_{n=1}^{N} a^*_n b_n.
$$

(3.6)

From this definition it follows that

$$
\langle \phi | \psi \rangle = \langle \psi | \phi \rangle^*.
$$

(3.7)

The length of a vector is called the “norm” in a general context and written $\| \psi \|$. As with ordinary vectors, the norm of a state vector in quantum mechanics is the square root of the inner product with itself, i.e.

$$
\| \psi \| = (\langle \psi | \psi \rangle)^{1/2} = \left( \sum_{n=1}^{N} |c_n|^2 \right)^{1/2} \quad \iff \quad |a| = (\vec{a} \cdot \vec{a})^{1/2} = \left( \sum_{i=1}^{n} |a_i|^2 \right)^{1/2}.
$$

(3.8)

As we shall see later, in quantum mechanics state vectors must have unit norm.

Orthogonality. Two state vectors are said to be orthogonal if their inner product is zero:

$$
\langle \phi | \psi \rangle = \langle \psi | \phi \rangle = 0, \quad \iff \quad \vec{a} \cdot \vec{b} = \vec{b} \cdot \vec{a} = 0.
$$

(3.9)

We choose basis states $|n \rangle$ which are normalized and orthogonal (orthonormal) so

$$
\langle n | m \rangle = \delta_{nm}, \quad \iff \quad \vec{e}_n \cdot \vec{e}_m = \delta_{nm}.
$$

(3.10)

From now on we will proceed with the formulation of quantum mechanics without giving the corresponding equations for ordinary (complex) vectors.

It will be useful to rewrite Eq. (3.2) for a linear superposition in a different way. Starting with Eq. (3.2),

$$
|\psi \rangle = \sum_{n=1}^{N} c_n |n \rangle,
$$

(3.11)

we take the inner product of both sides with the dual of one of the basis states, $|m \rangle$ say. Using the orthonormality property in Eq. (3.10) gives us

$$
c_m = \langle m | \psi \rangle,
$$

(3.12)
3.2 PHASES

and so we can rewrite Eq. (3.11) as

$$|\psi\rangle = \sum_{n=1}^{N} |n\rangle \langle n| \psi\rangle.$$  \hspace{1cm} (3.13)

This shows us that

$$\sum_{n=1}^{N} |n\rangle \langle n| = 1,$$  \hspace{1cm} (3.14)

the identity matrix. Equation (3.14) is sometimes called a completeness relation. A single term in this sum, $|n\rangle \langle n|$ is an $N \times N$ matrix with all elements 0 except that the $n$-th diagonal element is 1.

To make this more concrete consider the following example of a 2-state system, i.e. one qubit,

$$|\psi\rangle' = \begin{pmatrix} 1 \\ 2i \end{pmatrix}.$$  \hspace{1cm} (3.15)

This is not normalized because the norm is

$$||\psi'\|| = \sqrt{1^2 + 2^2} = \sqrt{5}.$$  \hspace{1cm} (3.16)

To get a valid quantum state it must be properly normalized so we divide by the norm. Hence

$$|\psi\rangle = \frac{1}{\sqrt{5}} |\psi\rangle' = \frac{1}{\sqrt{5}} \begin{pmatrix} 1 \\ 2i \end{pmatrix}.$$  \hspace{1cm} (3.17)

is a valid quantum state. To get the dual state vector we take the complex conjugate of the transpose, so

$$\langle \psi| = \frac{1}{\sqrt{5}} \begin{pmatrix} 1 \\ -2i \end{pmatrix}.$$  \hspace{1cm} (3.18)

Suppose we also have a second state,

$$|\phi\rangle = \frac{1}{\sqrt{5}} \begin{pmatrix} 2 \\ -i \end{pmatrix},$$  \hspace{1cm} (3.19)

which we see is normalized because $\sqrt{2^2 + (-i)^2} = \sqrt{5}$. What then is the inner product $\langle \psi|\phi\rangle$? We have

$$\langle \psi|\phi\rangle = \frac{1}{5} \begin{pmatrix} 1 & -2i \end{pmatrix} \begin{pmatrix} 2 \\ -i \end{pmatrix} = \frac{1}{5} (1 \cdot 2 + (-2i) \cdot (-i)) = 0,$$  \hspace{1cm} (3.20)

so $|\psi\rangle$ and $|\phi\rangle$ are actually orthogonal. In this example we had to be careful with the factors of $i$ because a complex conjugate is taken when we form the dual vector (which we need to get the inner product with another vector).

3.2 Phases

At this point it is convenient to discuss an important topic, namely phases. Suppose we have a 2-state system with complex amplitudes, which we write in polar form as

$$|\psi\rangle = r_0 e^{i\theta_0} |0\rangle + r_1 e^{i\theta_1} |1\rangle,$$  \hspace{1cm} (3.21)

where $r_0^2 + r_1^2 = 1$ for normalization. Let’s take out the factor of $e^{i\theta_0}$, so

$$|\psi\rangle = e^{i\theta_0} \left( r_0 |0\rangle + r_1 e^{i(\theta_1 - \theta_0)} |1\rangle \right).$$  \hspace{1cm} (3.22)
We call $\theta_0$ the overall phase which turns out to have no physical significance, while $\theta_1 - \theta_0$ is the relative phase (of basis states $|1\rangle$ and $|0\rangle$) which is important because it gives rise to interference. It is crucial to understand the difference between global phase and relative phase. **States which differ only in the overall phase are physically identical.**

For example,

$$|\psi_1\rangle = \frac{1}{\sqrt{2}} \begin{pmatrix} 1 \\ -1 \end{pmatrix}, \quad |\psi_2\rangle = \frac{1}{\sqrt{2}} \begin{pmatrix} 1 \\ 1 \end{pmatrix},$$

describe the same state because one is just the negative of the other. By contrast,

$$|\psi'_1\rangle = \frac{1}{\sqrt{2}} \begin{pmatrix} 1 \\ 1 \end{pmatrix}, \quad |\psi'_2\rangle = \frac{1}{\sqrt{2}} \begin{pmatrix} 1 \\ -1 \end{pmatrix},$$

describe different states because the relative phase of $|1\rangle$ and $|0\rangle$ is different in the two cases ($0$ for $|\psi'_1\rangle$ and $\pi$ for $|\psi'_2\rangle$).

To make sure we haven’t forgotten it, let’s reiterate (with a bit more math jargon) the first Ansatz of quantum mechanics which we stated at the beginning of this section:

**Ansatz 1:** The state of a quantum system is a vector in a complex vector space (technically a Hilbert space though we won’t need that level of mathematical sophistication here).

### 3.3 Observables

How is all this abstract stuff about complex vectors related to the real world, i.e. to quantities that we can measure.

The answer is that an observable quantity will be an operator, $\hat{O}$ say, acting on these vectors. The “hat” symbol “ $\hat{\ }$ ” indicates an operator, though for simplicity of notation we will usually omit the hat when context makes clear that we are dealing with an operator. In terms of components, operators will be represented by matrices.

An operator acting on a state vector gives another state vector, so

$$\hat{O}|\psi\rangle = |\phi\rangle,$$

A crucial point is that operators in quantum mechanics are *linear*, so

$$\hat{O} (a|\psi\rangle + b|\phi\rangle ) = a\hat{O}|\psi\rangle + b\hat{O}|\phi\rangle.$$

This brings us to the second Ansatz of quantum mechanics:

**Ansatz 2:** Observables are represented by linear Hermitian operators. The result of the measurement is one of the eigenvalues of the corresponding operator $\hat{O}$. After the measurement, the system is in the eigenstate corresponding to the measured eigenvalue.

Why is it assumed that quantity which can be measured is represented by a *Hermitian* operator? The answer is that the eigenvalues of a Hermitian operator (matrix) are guaranteed to be real, and we know that the results of a measurement must be real.

We now discuss how to represent operators as a matrix in a basis. This will also make us more familiar with the Dirac notation. We take orthonormal basis vectors $|n\rangle$ which have the orthonormality property $\langle m|n\rangle = \delta_{mn}$. In terms of components, $|n\rangle$ will be a column vector with the $n$-th entry equal
to 1 and all the others zero. In other words
\[
|n\rangle = \begin{pmatrix} \langle n \rangle_{1} \cr \vdots \cr \langle n \rangle_{n} \end{pmatrix}.
\]  
\hspace{1cm} (3.27)

Consider the action of an operator \( A \) on one of the basis vectors \( |n\rangle \). It will give a linear combination of the basis vectors.
\[
\begin{pmatrix} A_{11} & A_{12} & \cdots & A_{1n} & \cdots & A_{1,N-1} & A_{1N} \\
A_{21} & A_{22} & \cdots & A_{2n} & \cdots & A_{2,N-1} & A_{2N} \\
\vdots & \vdots & \ddots & \vdots & \ddots & \vdots & \vdots \\
A_{n1} & A_{n2} & \cdots & A_{nn} & \cdots & A_{n,N-1} & A_{nN} \\
A_{N-1,1} & A_{N-1,2} & \cdots & A_{N-1,n} & \cdots & A_{N-1,N-1} & A_{N-1,N} \\
A_{N1} & A_{N2} & \cdots & A_{Nn} & \cdots & A_{N,N-1} & A_{NN} \end{pmatrix} \begin{pmatrix} 0 \\
0 \\
\vdots \\
1 \\
0 \\
0 \end{pmatrix} = \begin{pmatrix} c_{1} \\
c_{2} \\
\vdots \\
c_{n} \\
c_{N-1} \\
c_{N} \end{pmatrix}.
\]  
\hspace{1cm} (3.28)

We see that \( c_{k} \) is equal to the element of \( A \) on the \( k \)-th row and \( n \)-th column, i.e. \( A_{kn} \). We can therefore write Eq. (3.28) as
\[
A \langle n \rangle = \sum_{k} A_{kn} |k\rangle.
\]  
\hspace{1cm} (3.29)

Acting on the left with the dual vector \( \langle m | \) and using the orthonormality of the basis vectors, we get
\[
A_{mn} = \langle m | A | n \rangle,
\]  
\hspace{1cm} (3.30)

which is the connection between the usual suffix notation for an element of a matrix, \( A_{mn} \), and the Dirac notation for the same thing, \( \langle m | A | n \rangle \). They both refer to the \( m \)-th row and \( n \)-th column of the matrix \( A \).

Recall that the definition of the adjoint of a matrix is \( A^\dagger = (A^T)^* \). Hence, in Dirac notation,
\[
\langle m | A^\dagger | n \rangle = \langle n | A^\dagger | m \rangle^*.
\]  
\hspace{1cm} (3.31)

If \( A \) is Hermitian then it is equal to its adjoint so
\[
\langle m | A | n \rangle = \langle n | A | m \rangle^* \quad \text{ (for \( A \) Hermitian).}
\]  
\hspace{1cm} (3.32)

To gain still more familiarity with the Dirac notation consider \( \langle \phi | A | \psi \rangle \). If we write this out in components in some basis, then \( |\psi\rangle \) is a column vector, \( A \) is a matrix and \( \langle \phi | \) is a row vector, i.e. we have

\[
\langle \phi | A | \psi \rangle = \begin{pmatrix} \phi_{1}^* & \phi_{2}^* & \cdots & \phi_{N}^* \end{pmatrix} \begin{pmatrix} A_{11} & A_{12} & \cdots & A_{1N} \\
A_{21} & A_{22} & \cdots & A_{2N} \\
\vdots & \vdots & \ddots & \vdots \\
A_{N1} & A_{N2} & \cdots & A_{NN} \end{pmatrix} \begin{pmatrix} \psi_{1} \\
\psi_{2} \\
\vdots \\
\psi_{N} \end{pmatrix},
\]  
\hspace{1cm} (3.33)

in an obvious notation. The multiplication can be done either by acting with \( A \) on \( |\psi\rangle \) to get \( |A\psi\rangle \) and then taking the inner product with \( \langle \phi | \), or by acting with \( A \) to the left on \( \langle \phi | \) and then taking the inner product with \( |\psi\rangle \). But what does acting with \( A \) to the left on \( \langle \phi | \) mean? Let’s suppose that
\[
\langle \phi | A = \langle \mu |.
\]  
\hspace{1cm} (3.34)
Then we have

\[
\begin{pmatrix}
\phi_1^* & \phi_2^* & \cdots & \phi_N^*
\end{pmatrix}
\begin{pmatrix}
A_{11} & A_{12} & \cdots & A_{1N} \\
A_{21} & A_{22} & \cdots & A_{2N} \\
\vdots & \vdots & \ddots & \vdots \\
A_{N1} & A_{N2} & \cdots & A_{NN}
\end{pmatrix}
= 
\begin{pmatrix}
\mu_1^* & \mu_2^* & \cdots & \mu_N^*
\end{pmatrix}.
\]

Evaluating components gives

\[
\mu_m^* = \sum_k \phi_k^* A_{km}.
\]

This can be rearranged as

\[
\mu_m = \sum_k \phi_k A_{km}^*
= \sum_k (A^T)^*_m \phi_k = \sum_k A^\dagger_{mk} \phi_k,
\]

or, for the vector as a whole

\[
|\mu\rangle = A^\dagger |\phi\rangle.
\]

which is equivalent to Eq. (3.34). Hence the action of \(A\) acting to the left on \(|\phi\rangle\) can be written as

\[
\langle \phi | A = \langle A^\dagger \phi |.
\]

Summarizing, we see that in \(\langle \phi | A |\psi\rangle\), the operator \(A\) can be considered to act either to the left or the right as follows:

\[
\langle \phi | A |\psi\rangle = \langle (A^\dagger \phi) |\psi\rangle = \langle \phi | (A \psi)\rangle.
\]

In quantum mechanics \(A\) will commonly be a Hermitian operator (since observables are represented by Hermitian operators) for which \(A^\dagger = A\), so \(A\) acts equally to the right and to the left as follows:

\[
\langle \phi | A |\psi\rangle = \langle (A \phi) |\psi\rangle = \langle \phi | (A \psi)\rangle \quad \text{(for } A\text{ Hermitian)}.
\]

When dealing with standard vectors, we know that we can work with different sets of bases rotated with respect to each other. In quantum mechanics, too, it will be convenient to represent state vectors in terms of different bases, transformed with respect to each other.

A common basis for a single qubit comprises the states \(|0\rangle\) and \(|1\rangle\), and in this basis the Pauli operator \(Z\) is diagonal, see Eq. (2.19). This is the most common basis used in quantum computing, and is called the computational basis. Since \(Z\) is diagonal in this basis the eigenvectors of \(Z\) are the basis vectors. For this reason the computational basis is sometimes called the \(Z\)-basis.

We will also need to consider other bases, one of the most common being the \(X\)-basis, i.e. the basis in which \(X\) (see Eq. (2.19)) is diagonal. You should be able to show that the eigenvalues of \(X\) are +1 and −1, with corresponding eigenvectors, called \(|+\rangle\) and \(|-\rangle\) (sometimes \(|0_x\rangle\) and \(|1_x\rangle\)), given by

\[
|0_x\rangle \equiv |+\rangle = \frac{1}{\sqrt{2}} (|0\rangle + |1\rangle)
\]

\[
|1_x\rangle \equiv |-\rangle = \frac{1}{\sqrt{2}} (|0\rangle - |1\rangle).
\]

There is a linear relation between the new basis vectors and the old ones. Denoting the old basis vectors by Latin letters, e.g. \(|n\rangle\), and the new basis vectors by Greek letters, e.g. \(|\alpha\rangle\), we write

\[
|\alpha\rangle = \sum_n U_{\alpha n} |n\rangle.
\]
3.4. OUTER PRODUCT NOTATION

The new basis vectors must be orthonormal, like the old set, and this constrains the matrix of coefficients $U$ in a way that we will now determine. Taking the complex conjugate of Eq. (3.43) gives

$$\langle \beta | = \sum_k U_{\beta k}^* \langle k |.$$  \hspace{1cm} (3.44)

Taking the inner product of Eqs. (3.43) and (3.44) gives

$$\langle \beta | \alpha \rangle = \sum_{n,k} U_{\beta k}^* U_{\alpha n} \langle k | n \rangle = \sum_n U_{\beta n}^* U_{\alpha n} = \sum_n \delta_{\alpha \beta} = \delta_{\alpha \beta},$$ \hspace{1cm} (3.45)

where we used that $\langle k | n \rangle = \delta_{kn}$ to get the second line. However, $\langle \beta | \alpha \rangle = \delta_{\alpha \beta}$ and so we must have $UU^\dagger = 1$, the identity matrix. Thus the matrix of coefficients which transforms from one basis to another as in Eq. (3.43) must be unitary.

As an example, according to Eq. (3.42) the matrix which transforms from the $Z$-basis to the $X$ basis for one qubit is

$$U = \frac{1}{\sqrt{2}} \begin{pmatrix} 1 & 1 \\ 1 & -1 \end{pmatrix}.$$ \hspace{1cm} (3.46)

We can verify that this matrix is unitary by evaluating its inverse, or, more simply, by recalling that the rows of a unitary matrix are orthonormal vectors, and the same for the columns. By inspection, this is the case here.

3.4 Outer Product Notation

For orthonormal basis vectors, we have $\langle i | j \rangle = \delta_{ij}$. As a further exercise in familiarization with the Dirac notation, consider what we mean if we write the vector and the dual vector the other way round i.e. $|i\rangle \langle j |$, which is called an “outer product”. It is actually a matrix. By sandwiching it on the left and right by basis states we see that it is a matrix, whose entries are all zero except for the element on the $i$-th row and $j$-th column which is 1. In other words

$$|i\rangle \langle j | = \begin{pmatrix} 0 & 0 & 0 & \cdots & 0 & \cdots & 0 \\ 0 & 0 & 0 & \cdots & 0 & \cdots & 0 \\ \vdots & \vdots & \vdots & \ddots & \vdots & \ddots & \vdots \\ 0 & 0 & 0 & \cdots & 1 & \cdots & 0 \\ \vdots & \vdots & \vdots & \ddots & \vdots & \ddots & \vdots \\ 0 & 0 & 0 & \cdots & 0 & \cdots & 0 \\ 0 & 0 & 0 & \cdots & 0 & \cdots & 0 \end{pmatrix}.$$ \hspace{1cm} (3.47)

If $j = i$, then we have a 1 on the $i$-th diagonal element and 0 everywhere else. This is a projection operator on to state $i$, so we denote it by $P_i$, i.e.

$$P_i = |i\rangle \langle i |.$$ \hspace{1cm} (3.49)

One can see it is a projection operator because, if it acts on an arbitrary state $|\psi \rangle$, we have

$$P_i |\psi \rangle = |i \rangle \langle i | \psi \rangle,$$ \hspace{1cm} (3.50)
which is the amplitude $\langle i | \psi \rangle$ for $| \psi \rangle$ to be along $| i \rangle$, times the state $| i \rangle$.

Clearly $\sum_i P_i$ has 1 on all the diagonal elements and is zero otherwise, so it is the identity matrix, i.e.

$$\sum_i P_i \equiv \sum_i | i \rangle \langle i | = 1.$$  \tag{3.51}

Similarly, for an arbitrary state $| \phi \rangle$, the outer product $| \phi \rangle \langle \phi |$ is a projection operator on to state $| \phi \rangle$ since, if it acts on a state $| \psi \rangle$, the result is $| \phi \rangle \langle \phi | \psi \rangle$.

### 3.5 Functions of operators

We will need to evaluate functions of operators. For example what is $e^A$? In this case there is a convergent series expansion which can be used to evaluate the function;

$$e^A = 1 + A + \frac{A^2}{2!} + \frac{A^3}{3!} + \cdots.$$  \tag{3.52}

In some cases the infinite series can be evaluated in closed form. Consider for example $e^{cX}$ where $c$ is a constant and $X$ is given in Eq. (2.19). We have $X^2 = 1$ and so $X^3 = X^5 \cdots = X^{2n+1} \cdots = X$, while $X^2 = X^4 \cdots = X^{2n} \cdots = 1$. Hence

$$e^{cX} = 1 \left( 1 + \frac{c^2}{2!} + \frac{c^4}{4!} + \cdots \right) + X \left( c + \frac{c^3}{3!} + \frac{c^5}{5!} + \cdots \right),$$

$$= 1 \cosh c + X \sinh c = \begin{pmatrix} \cosh c & \sinh c \\ \sinh c & \cosh c \end{pmatrix}.$$  \tag{3.53}

More generally, when there is not a convergent expansion we can evaluate a function of an operator by diagonalizing it. Consider first a diagonal matrix,

$$D = \begin{pmatrix} \lambda_1 & 0 & \cdots & 0 \\ 0 & \lambda_2 & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & \lambda_N \end{pmatrix}.$$  \tag{3.54}

When multiplying $D$ by itself $n$ times, say, all that happens is each diagonal element is multiplied by itself $n$ times. Hence if $f(D)$ is some function of $D$,

$$f(D) = \begin{pmatrix} f(\lambda_1) & 0 & \cdots & 0 \\ 0 & f(\lambda_2) & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & f(\lambda_N) \end{pmatrix}.$$  \tag{3.55}

In general, a matrix $A$ is not already in diagonal form. However, we can diagonalize it by a similarity transform, see Eq. (2.30), which we repeat here:

$$A = SDS^{-1},$$  \tag{3.56}
where \( D \) is a diagonal matrix with the eigenvalues of \( A \) on the diagonal. Hence it follows that

\[
A^2 = SDS^{-1}S = SD^2S^{-1},
\]
\[
A^3 = SDS^{-1}SDS^{-1}S = SD^3S^{-1} , \quad \text{and so}
\]
\[
A^n = SD^nS^{-1}, \quad \text{and hence}
\]
\[
f(A) = Sf(D)S^{-1}
\]

which is the desired expression showing how to construct a function of a matrix from its eigenvalues and eigenvectors.

### 3.6 Measurements

Now we have to discuss in detail the vexed topic of measurement in quantum mechanics. The reason for using the term “vexed” will become clear later, especially in Chapter 7 when we discuss a famous thought experiment of Einstein, Podolsky and Rosen (EPR).

In a measurement, our delicate quantum system is brought into contact with a macroscopic experimental apparatus. Measurement is an irreversible process and as such has a special status in quantum mechanics.

Assume that the Hermitian operator \( A \) corresponding to the measured quantity of interest has eigenvalues \( \lambda_n \) and normalized eigenvectors \( |n\rangle \). Because \( A \) is Hermitian the eigenvalues are real. In addition, for a matrix of size \( N \) there are \( N \) orthogonal eigenvectors which can therefore be used as a basis. Hence, we can write the state of the system before measurement, \(|\psi\rangle\), as a linear superposition of the eigenvectors of \( A \),

\[
|\psi\rangle = \sum_n a_n |n\rangle.
\]

where the last line is from Eq. (3.13). We will call \( a_n \equiv \langle n|\psi\rangle \) the amplitude for \(|\psi\rangle\) to be in eigenstate \(|n\rangle\).

The measurement will give one of the eigenvalues, \( \lambda_n \), but which one? To answer this question, we need to add one more ingredient to our Ansatz 2, one which was first proposed by Born in a footnote in a 1926 paper, and which is therefore called the “Born rule”. This states that the probability, \( P(n) \), to get eigenvalue \( \lambda_n \) (and after the measurement to leave the system in eigenstate \(|n\rangle\)), is the square of the modulus of the amplitude \( a_n \), i.e.

\[
P(n) = |a_n|^2 \equiv |\langle n|\psi\rangle|^2 \equiv \langle \psi|n\rangle \langle n|\psi\rangle, \tag{3.59}
\]

where we used Eq. (3.12) and that \( \langle \psi|n\rangle = \langle n|\psi\rangle^\ast \). Since probabilities must add up to 1, it follows that state vectors in quantum mechanics must be normalized to unity, i.e.

\[
\sum_n |a_n|^2 = \sum_n |\langle n|\psi\rangle|^2 \equiv \sum_n \langle \psi|n\rangle \langle n|\psi\rangle = 1. \tag{3.60}
\]

We therefore have to complete our Ansatz 2 to include the probabilities of different results:
Ansatz 2': "Observables are represented by a linear Hermitian operator. Measurement of an observable corresponding to a (linear) Hermitian operator $\hat{O}$ gives one of the eigenvalues of $\hat{O}$. The probability of getting an eigenvalue is the square of the modulus of the amplitude for the state of the system to be in the corresponding eigenstate of $\hat{O}$. After the measurement, the system is in this eigenstate."

It is the fact that probabilities enter into the results of measurements that has led to a lot of "vexed" discussion. Your first reaction might be "What's the fuss? After all, don't probabilities enter in classical physics too? If one tosses a coin isn't the result is randomly heads or tails with equal probability?" Well, is it really random? If one could measure with sufficient precision the initial momentum and angular momentum of the coin, and integrate the equations of motion for its trajectory, including the effects of air resistance, to sufficient accuracy then one would be able to compute, with certainty, on which side it would land. The difficulty is that the coin toss has great sensitivity to the initial conditions, which means that if one changes the initial velocity by an immeasurably small amount the result changes. In other words, for all practical purposes (FAPP) a coin toss is random. Nonetheless, from a fundamental point of view it is not, since it is uniquely determined by the initial conditions. However, the situation in quantum mechanics is different since, as far as we know, probabilities enter in a fundamental way.

The most famous critic of probabilities being part of a fundamental theory of physics was Einstein, who had many discussions on the topic with Niels Bohr. As we shall see in our study of the EPR thought experiment in Chapter 7, despite Einstein's claim that "God doesn't play dice with the universe", quantum mechanics has been repeatedly vindicated.

We have said that after a measurement the system is left in eigenstate $|n\rangle$. Measurement therefore "projects" the initial state $|\psi\rangle$ on to $|n\rangle$. This is accomplished by the projection operator

$$\hat{P}_n = |n\rangle\langle n|$$

so

$$\hat{P}_n |\psi\rangle = |n\rangle\langle n| |\psi\rangle,$$

(no sum on $n$). The sum of the projection operators must add to the identity, i.e.

$$\sum_n \hat{P}_n \equiv \sum_n |n\rangle\langle n| = 1.$$

The fact that $\sum_n |n\rangle\langle n|$ can be replaced by the identity is called a "completeness" relation.

Note that the state in Eq. (3.62) is not normalized. If we continue to follow the system after the measurement then we need to multiply the state by $1/|\langle n|\psi\rangle|$, so it is again correctly normalized and the sum of probabilities of results of a future measurement will add to unity. We note that something similar is also done in classical statistics. If we have a sequence of measurements, and we know the result of the first one, then we can determine the "conditional probability" of subsequently measurements, conditioned on the result of the first measurement, and these conditional probabilities add to unity. In effect, this is what is done by multiplying a state by a constant to get its norm back to 1 after a measurement. We are then determining conditional probabilities for a subsequent measurement given the result of the first measurement.

Let's give a simple example of a measurement. Consider one qubit in state $|\psi\rangle = \frac{1}{\sqrt{2}}(|0\rangle + |1\rangle)$ and measure $Z$. The eigenstates of $Z$ are $|0\rangle$ and $|1\rangle$ with eigenvalues $+1$ and $-1$ respectively. Hence the results of a measurement of $Z$ are

$$+1, \quad \text{prob.} \left(\frac{1}{\sqrt{2}}\right)^2 = \frac{1}{2},$$

$$-1, \quad \text{prob.} \left(\frac{1}{\sqrt{2}}\right)^2 = \frac{1}{2}.$$
Now suppose that we measure $X$. The eigenstates of $X$ are shown in Eqs. (2.24) and (2.26) to be $\frac{1}{\sqrt{2}}(|0\rangle \pm |1\rangle)$. Hence $|\psi\rangle$ is the eigenstate with eigenvalue +1, so the result of the measurement of $X$ is +1 with 100% probability.

We see that if the initial state is an eigenstate of the operator being measured, then the result will, with certainty, be the corresponding eigenvalue, and the state will remain unchanged after the measurement. However, if this is not the case, i.e. if the initial state is in a superposition of eigenstates of the measurement operator, then (i) the result of the measurement will take one of several values with appropriate probabilities, and (ii) the measurement changes the state, leaving it in the eigenstate corresponding to the eigenvalue which is measured.

### 3.7 Statistics of Measurements

If we prepare many identical copies of the system and measure each of them what can we say about the statistics of the measured values. First of all, what would be the mean of the measurements $\langle A \rangle$? We have

$$
\langle A \rangle = \sum_n P(n) \lambda_n \\
= \sum_n |\langle n | \psi \rangle|^2 \lambda_n = \sum_n \langle \psi | n \rangle \lambda_n \langle n | \psi \rangle \\
= \sum_n \langle \psi | A | n \rangle \langle n | \psi \rangle \\
= \langle \psi | A | \psi \rangle . 
$$

(3.65)

where use used Eq. (3.59) to get the second line, we used that $A | n \rangle = \lambda_n | n \rangle$ to get the third line, and Eq. (3.63) to get the last line. The final result, $\langle \psi | A | \psi \rangle$ is called the “expectation value” of $A$ in state $|\psi\rangle$.

In addition to the average result we are also often interested in the scatter about the average. This is characterized by the standard deviation defined by

$$
\Delta A = \left( \left\langle (A - \langle A \rangle)^2 \right\rangle \right)^{1/2},
$$

(3.66)

which is the root mean square deviation about the mean. It can be expressed in a slightly simpler form since

$$
\left\langle (A - \langle A \rangle)^2 \right\rangle = \langle A^2 \rangle - 2\langle A \rangle \langle A \rangle + \langle A \rangle^2 \\
= \langle A^2 \rangle - 2\langle A \rangle^2 + \langle A \rangle^2 \\
= \langle A^2 \rangle - \langle A \rangle^2,
$$

(3.67)

so

$$
\Delta A = \left( \langle A^2 \rangle - \langle A \rangle^2 \right)^{1/2}
$$

(3.68)

We will also call $\Delta A$ the uncertainty in $A$.

Let’s illustrate this with the example we considered just above, namely $|\psi\rangle = \frac{1}{\sqrt{2}}(|0\rangle + |1\rangle)$. If we measure $Z$ we have

$$
\langle Z \rangle = \frac{1}{2} \begin{pmatrix} 1 & 1 \\ 1 & 0 \end{pmatrix} \begin{pmatrix} 1 \\ 0 \end{pmatrix} \begin{pmatrix} 1 \\ 1 \end{pmatrix} = 0.
$$

(3.69)

This agrees with our previous discussion where we found +1 and −1 with equal probability. We also have $Z^2 = 1$ and, since the average of 1 is always 1,

$$
\langle Z^2 \rangle = 1,
$$

(3.70)
\[ \Delta Z = \left( \langle Z^2 \rangle - \langle Z \rangle^2 \right)^{1/2} = 1. \]  

(3.71)

For a measurement of \( X \) we already showed that \( |\psi\rangle \) is an eigenstate with eigenvalue 1 and so the measured value is always 1. If we use Eqs. (3.65) and (3.68), we recover the expected results, \( \langle X \rangle = 1 \) and \( \Delta X = 0 \).

### 3.8 Composite Systems

So far, we have described states of just a single qubit. How should we describe states of the many qubits which we will need for a quantum computer? Suppose, as an example, we have two qubits \( A \) and \( B \). We can label the states of qubit \( A \) by \( |0_A\rangle \) and \( |1_A\rangle \), and similarly the states of qubit \( B \) by \( |0_B\rangle \) and \( |1_B\rangle \). A state of both qubits is written as a “tensor product”, e.g. \( |0_A\rangle \otimes |1_B\rangle \), which in this example indicates that qubit \( A \) is in state \( |0\rangle \) and qubit \( B \) is in state \( |1\rangle \).

This notation is heavy so we will usually write the same state more compactly as \(|01\rangle\), or even more concisely as \(|0\rangle\) provided a specification of the order of the qubits has been given. In this notation, the four possible states of two qubits are

\[ |00\rangle, \quad |01\rangle, \quad |10\rangle, \quad |11\rangle. \]  

(3.72)

Note that the label of each state is a number in binary notation from 0 to 3. This provides an even more compact notation, which is particularly convenient when the number of qubits is large, namely \(|x\rangle_2\), where \( x = 0, 1, 2 \) or 3. It is necessary to indicate the number of qubits by a subscript on the bracket to avoid ambiguity. For example just writing a state as \(|2\rangle\) we wouldn’t know if it is state \(|10\rangle\) for 2 qubits, or \(|010\rangle\) for 3 qubits and so on. An exception to this will be states \(|0\rangle \) and \(|1\rangle \) (without subscript) which always refer to the 1-qubit basis states. The four states in Eq. (3.72) can therefore also be written as

\[ |0\rangle_2, \quad |1\rangle_2, \quad |2\rangle_2, \quad |3\rangle_2. \]  

(3.73)

Similarly for three qubits, we can specify the 8 possible states by \(|x\rangle_3\) where \( x = 0, 1, \cdots, 7 \), and for \( n \) qubits the \( 2^n \) states are indicated by \(|x\rangle_n\), where \( x = 0, 1, \cdots, 2^n - 1 \). We see that to use this convenient binary notation we need to label the states starting from 0 rather than 1. The last state then has label \( 2^n - 1 \).

We need to be familiar with these ways of labeling multi-qubit states.

Next we discuss matrix representations of operators on multiple qubits, and we take as an example, the case of two qubits. An operator acting on the space of two qubits is a \( 4 \times 4 \) matrix. We will write the four basis states as \(|00\rangle, \quad |01\rangle, \quad |10\rangle, \quad |11\rangle\). Consider an operator where \( X \) acts on the first (left hand) qubit and the identity \( \mathbb{1} \) acts on the second (right hand) qubit. The 2-qubit operator is a tensor product of the 1-qubit operators, i.e. \( X \otimes \mathbb{1} \). Its action on the four basis states is as follows:

\[
\begin{align*}
X \otimes \mathbb{1} |00\rangle &= |10\rangle, \\
X \otimes \mathbb{1} |01\rangle &= |11\rangle, \\
X \otimes \mathbb{1} |10\rangle &= |00\rangle, \\
X \otimes \mathbb{1} |11\rangle &= |01\rangle,
\end{align*}
\]  

(3.74)

so its matrix representation is

\[
X \otimes \mathbb{1} = \begin{pmatrix}
|00\rangle & |01\rangle & |10\rangle & |11\rangle \\
0 & 0 & 1 & 0 \\
0 & 0 & 0 & 1 \\
1 & 0 & 0 & 0 \\
0 & 1 & 0 & 0
\end{pmatrix} = \begin{pmatrix}
0 & 1 \\
1 & 0
\end{pmatrix},
\]  

(3.75)
where in the last expression each entry is a $2 \times 2$ block. Note how this block structure reflects the operators in the tensor product on the left of the expression. The $2 \times 2$ block structure is that of $X$ while each block is made up of the identity $I$.

As a second example consider $X \otimes Z$. We have

\[
X \otimes Z|00\rangle = |10\rangle, \\
X \otimes Z|01\rangle = -|11\rangle, \\
X \otimes Z|10\rangle = |00\rangle, \\
X \otimes Z|11\rangle = -|01\rangle,
\]

so its matrix representation is

\[
X \otimes Z = \begin{pmatrix}
|00\rangle & |01\rangle & |10\rangle & |11\rangle \\
0 & 0 & 1 & 0 \\
0 & 0 & 0 & -1 \\
1 & 0 & 0 & 0 \\
0 & -1 & 0 & 0
\end{pmatrix} = \begin{pmatrix}
0 & Z \\
Z & 0
\end{pmatrix}.
\]

Again notice how the block structure in the last expression reflects the operators in the tensor product.

### 3.9 Generalized Born Rule

In Sec. 3.6 we gave the standard physics text book discussion of measurement in quantum mechanics. For quantum computing we need to extend this to deal with situations involving multiple qubits where we measure only some of the qubits and we need to know the state of the remaining qubits after the measurement. As a simple example, suppose we have 2 qubits $A$ and $B$, in a state

\[
|\psi\rangle = a_0|00\rangle + a_1|01\rangle + a_2|10\rangle + a_3|11\rangle,
\]

where the left qubit is $A$ and the right qubit is $B$. Because the state has to be normalized we need $|a_0|^2 + |a_1|^2 + |a_2|^2 + |a_3|^2 = 1$. Suppose we measure $Z$ for qubit-$A$, the left qubit. We want to know what are the possible measurement results, what are the probabilities of the different results, and in what state $B$ is after the measurement.

We rewrite Eq. (3.78), grouping together all the terms where qubit $A$ is $|0\rangle$ (more generally an eigenstate of the operator acting on $A$), and all the terms where qubit $A$ is $|1\rangle$ (the other eigenstate):

\[
|\psi\rangle = \alpha_0|0_A\rangle|\phi_{0,B}\rangle + \alpha_1|1_A\rangle|\phi_{1,B}\rangle,
\]

where

\[
|\alpha_0|^2 = |a_0|^2 + |a_1|^2, \\
|\alpha_1|^2 = |a_2|^2 + |a_3|^2,
\]

so $|\alpha_0|^2 + |\alpha_1|^2 = 1$.

\[
|\phi_{0,B}\rangle = \frac{1}{\alpha_0} (a_0|0_B\rangle + a_1|1_B\rangle), \\
|\phi_{1,B}\rangle = \frac{1}{\alpha_1} (a_2|0_B\rangle + a_3|1_B\rangle).
\]

Note that $|\phi_{0,B}\rangle$ and $|\phi_{1,B}\rangle$ are normalized, and, because of Eq. (3.82), so is $|\psi\rangle$ in Eq. (3.79). The inner product of $|\phi_0\rangle$ and $|\phi_1\rangle$ is

\[
\langle \phi_0 | \phi_1 \rangle = \frac{a_0^* a_2 + a_1^* a_3}{\sqrt{|a_0|^2 + |a_1|^2} \sqrt{|a_2|^2 + |a_3|^2}},
\]
and there is no reason for this to be zero in general (we drop the subscript \( B \) when there is no need to specify which qubit is involved). Hence \( |\phi_0\rangle \) and \( |\phi_1\rangle \) are not necessarily orthogonal.

The natural extension of the Born rule, called the “generalized Born” rule, is that, when the two qubits are in the state given in Eq. \( (3.79) \), the possible results of the measurement of \( Z \) on qubit \( A \), are

\[
\begin{align*}
\text{result } +1, & \quad \text{probability } |\alpha_0|^2 , \quad \text{final state } |0\rangle_A|\phi_{0,B}\rangle, \\
\text{result } -1, & \quad \text{probability } |\alpha_1|^2 , \quad \text{final state } |1\rangle_A|\phi_{1,B}\rangle.
\end{align*}
\]

(3.86)

It is straightforward to generalize this result to an arbitrary situation in which there are \( n + m \) qubits, \( n \) of which are measured and we want to know the possible final states of the remaining \( m \) qubits after the measurement, and to arbitrary measurement operators.

### 3.10 The Uncertainty Principle

Now we come to a key concept in quantum mechanics, the *uncertainty principle*. We shall see that some variables are incompatible with each other, which means that one can not have definite values for both of them in *any* state. The important quantity to see if two operators, \( A \) and \( B \) say, are compatible is their commutator

\[
[A, B] \equiv AB - BA.
\]

(3.87)

If \([A, B] \neq 0\) then it is shown in linear algebra texts that \( A \) and \( B \) have different eigenvectors. We have already noted that we only get a definite value for some operator if the state is in an eigenstate of that operator. Hence, if \([A, B] \neq 0\), so \( A \) and \( B \) have different eigenvectors, there is no state which will give a definite value for both of them.

As an example of a commutator consider \( X \) and \( Z \). We have

\[
[Z, X] = \begin{pmatrix} 1 & 0 \\ 0 & -1 \end{pmatrix} \begin{pmatrix} 0 & 1 \\ 1 & 0 \end{pmatrix} - \begin{pmatrix} 0 & 1 \\ 1 & 0 \end{pmatrix} \begin{pmatrix} 1 & 0 \\ 0 & -1 \end{pmatrix} = \begin{pmatrix} 0 & 1 \\ 1 & 0 \end{pmatrix} - \begin{pmatrix} 0 & -1 \\ 1 & 0 \end{pmatrix} = 2 \begin{pmatrix} 0 & 1 \\ -1 & 0 \end{pmatrix} = 2iY,
\]

(3.88)

where \( Y \) is defined in Eq. \( (2.19) \). Since the commutator is non-zero it is impossible to find a state which is simultaneous eigenstate of both \( X \) and \( Z \) and so either \( \Delta X \) or \( \Delta Z \), or both, must be non-zero.

An important inequality involving the uncertainties \( \Delta A \) and \( \Delta B \) of two operators in a state \( |\psi\rangle \) is

\[
(\Delta A \Delta B)_{\psi} \geq \frac{1}{2} \left| \langle [A, B] \rangle_{\psi} \right|,
\]

(3.89)

which is known as the Heisenberg uncertainty principle. We shall not prove this result. The most famous case of the uncertainty principle is for \( A = x \), the position of a particle, and \( B = p \), its momentum, for which the commutator is a constant \( 2i\hbar \) so

\[
\Delta x \Delta p \geq \frac{\hbar}{2}.
\]

(3.90)

However, this particular version of the uncertainty principle does not play a role in quantum computing which is concerned with (discrete) 2-state systems, rather than (continuous) trajectories of particles.

\(^2\hbar \) is Planck's constant divided by \( 2\pi \). It is of paramount importance in physics but does not play a role in the theory of quantum computation.
3.11 Time Evolution of Quantum States

So far, we have described fixed quantum states. Now we need to discuss how they evolve with time. If the state at an initial time is $|\psi\rangle$ and the state at a later time is $|\psi'\rangle$, then, by assumption, there is a linear relation between the two, so

$$|\psi'\rangle = U|\psi\rangle,$$  \hspace{1cm} (3.91)

for some linear operator $U$. The normalization condition must be preserved so $\langle \psi'|\psi' \rangle = \langle \psi|\psi \rangle = 1$. This provides a constraint on the form of $U$ as we will now show. The equation corresponding to Eq. (3.91) for the dual vector $\langle \psi'|$ is

$$\langle \psi'| = \langle \psi|U^\dagger.$$  \hspace{1cm} (3.92)

To see this compare Eqs. (3.38) and (3.34) and note that $(A^\dagger)^\dagger = A$. Combining Eqs. (3.91) and (3.92) we find

$$\langle \psi'|\psi' \rangle = \langle \psi|U^\dagger U|\psi \rangle.$$  \hspace{1cm} (3.93)

Since we must have $\langle \psi'|\psi' \rangle = \langle \psi|\psi \rangle (= 1)$ for any initial state $|\psi\rangle$ it follows that

$$U^\dagger U = 1,$$  \hspace{1cm} (3.94)

so $U$ has to be unitary.

In quantum computing we change the state of the qubits by a sequence of discrete unitary transformations. Note that for a unitary operator $U^{-1} = U^\dagger$, and $U^\dagger$ is well defined, so the inverse transformation, which acts on the final state and converts it to the initial state, exists. Thus quantum transformations are reversible. The exception is measurement, in which the quantum system is coupled to a macroscopic, external apparatus which leads to an irreversible change. As we shall see, standard classical gates, which manipulate the bits in a classical computer, are irreversible. The necessity of doing reversible operations in a quantum computer will be a major difference compared with a classical computer.

In a quantum computer, as noted above, we act on the qubits with a series of discrete unitary operations. These come from the effect of some operation acting for a finite amount of time.

Microscopically, quantum states evolve continuously with time, and we will finish this Chapter with a brief discussion of continuous time evolution in quantum mechanics (even though it will not be needed in the rest of the course). Time evolution is determined by the Hamiltonian (energy), $\mathcal{H}$ a Hermitian operator, according to Ansatz 3:

**Ansatz 3:** The time dependence of a state is given by Schrödinger’s equation

$$i\hbar \frac{\partial}{\partial t} |\psi(t)\rangle = \mathcal{H} |\psi(t)\rangle.$$  \hspace{1cm} (3.95)

The constant $\hbar$ is mentioned in footnote 2 on page 28 and will be set to unity in the rest of this section.

Assuming that $\mathcal{H}$ does not change with time, we can integrate Eq. (3.95) to get

$$|\psi(t)\rangle = U(t)|\psi(0)\rangle,$$  \hspace{1cm} (3.96)

where

$$U(t) = e^{-i\mathcal{H}t}.$$  \hspace{1cm} (3.97)

Since $\mathcal{H}$ is Hermitian we can show that $U$ is unitary by the following argument. To get the adjoint of $U$ we take its complex conjugate and replace any operators in the expression for $U$ by their adjoint. Since $\mathcal{H}$ is self-adjoint (Hermitian) we have

$$U^\dagger(t) = e^{i\mathcal{H}t},$$  \hspace{1cm} (3.98)
from which one sees that
\[ U^\dagger(t)U(t) = e^{i\mathcal{H}t}e^{-i\mathcal{H}t} = e^{i\mathcal{H}t-i\mathcal{H}t} = 1, \] (3.99)
so \( U \) is unitary as required. Note that if we have operators in exponentials which don’t commute, we can’t manipulate them as we do with ordinary numbers. For example \( e^Ae^B \) does not equal \( e^{A+B} \) unless \([A,B]=0\). However, here both \( A \) and \( B \) are proportional to \( \mathcal{H} \) which commutes with itself, so combining the exponentials as done in Eq. (3.99) is valid.
Chapter 4

General state of a qubit, no-cloning theorem, entanglement and Bell states

4.1 General qubit states

The following $2 \times 2$ matrices, called Pauli matrices, acting on the states of a single qubit will be important in the rest of the course:

$$
X \equiv \sigma_x = \begin{pmatrix} 0 & 1 \\ 1 & 0 \end{pmatrix},
$$

(4.1a)

$$
Y \equiv \sigma_y = \begin{pmatrix} 0 & -i \\ i & 0 \end{pmatrix},
$$

(4.1b)

$$
Z \equiv \sigma_z = \begin{pmatrix} 1 & 0 \\ 0 & -1 \end{pmatrix}.
$$

(4.1c)

In the physics literature the notation used is $\sigma_x$ etc., but in this course we shall use the quantum computing notation: $X$, $Y$, and $Z$. As shown in Sec. 2.3 an arbitrary $2 \times 2$ matrix can be written as a linear combination of the three Pauli matrices plus the $2 \times 2$ identity matrix.

These matrices are Hermitian, and have eigenvalues $\pm 1$, see Sec. 2.3.

If the qubit is the spin of an electron, then the eigenstate with $Z = 1$ has spin along the $+z$ direction, and similarly the eigenstate with $Y = 1$ has spin along the $+y$ direction, and the eigenstate with $X = 1$ has spin along the $+x$ direction. Also, the eigenstate with $Z = -1$ has the spin pointing in the $-z$ direction, and similarly for $X = -1$ and $Y = -1$.

It is useful to consider eigenstates for the spin aligned along a general direction with polar angles $\theta$ and $\phi$, indicated by a unit vector $\hat{n}$ where

$$
\hat{n} = (\sin \theta \cos \phi, \sin \theta \sin \phi, \cos \theta),
$$

(4.2)

so $n_x = \sin \theta \cos \phi$ etc. In other words we compute the eigenvalues and eigenvectors of $\vec{\sigma} \cdot \hat{n}$. We have

$$
\vec{\sigma} \cdot \hat{n} = \begin{pmatrix} n_z & n_x - in_y \\ n_x + in_y & -n_z \end{pmatrix}
$$

(4.3)

so the eigenvalues are given by

$$
\begin{vmatrix}
  n_z - \lambda & n_x - in_y \\
  n_x + in_y & -n_z - \lambda
\end{vmatrix} = 0.
$$

(4.4)

Expanding the determinant, and using that $n_x^2 + n_y^2 + n_z^2 = 1$, we find the eigenvalues to be

$$
\lambda = \pm 1.
$$

(4.5)
Thus, the eigenvalues are not only $\pm 1$ when measured along the Cartesian directions, but take the same values along any direction.

Next we look at the eigenvectors. First the eigenvector for eigenvalue $+1$ is

$$|0_{\hat{n}}\rangle = \begin{pmatrix} a \\ b \end{pmatrix} \quad (4.6)$$

where

$$\begin{pmatrix} \cos \theta & \sin \theta e^{-i\phi} \\ \sin \theta e^{i\phi} & -\cos \theta \end{pmatrix} \begin{pmatrix} a \\ b \end{pmatrix} = \begin{pmatrix} a \\ b \end{pmatrix}, \quad (4.7)$$

where we used Eqs. (4.2) and (4.3). Writing out the two equations we get

$$\sin \theta e^{-i\phi} b = a(1 - \cos \theta), \quad (4.8a)$$

$$\sin \theta e^{i\phi} a = b(1 + \cos \theta). \quad (4.8b)$$

Both these equations are satisfied by

$$b \cos \frac{\theta}{2} = a e^{i\phi} \sin \frac{\theta}{2}, \quad (4.9)$$

(recall the expressions for sines and cosines of double angles). We require the state to be normalized, i.e. $|a|^2 + |b|^2 = 1$, so we get

$$|0_{\hat{n}}\rangle = \begin{pmatrix} \cos \frac{\theta}{2} \\ e^{i\phi} \sin \frac{\theta}{2} \end{pmatrix}, \quad (4.10)$$

or equivalently, in Dirac notation,

$$|0_{\hat{n}}\rangle = \cos \frac{\theta}{2} |0\rangle + e^{i\phi} \sin \frac{\theta}{2} |1\rangle. \quad (4.11a)$$

A similar calculation gives the eigenstate corresponding to eigenvalue $-1$ to be

$$|1_{\hat{n}}\rangle = -\sin \frac{\theta}{2} |0\rangle + e^{i\phi} \cos \frac{\theta}{2} |1\rangle. \quad (4.11b)$$

It is straightforward to see that the states in Eqs. (4.11) are normalized, i.e.

$$\langle 0_{\hat{n}}|0_{\hat{n}}\rangle = 1, \quad (0_{\hat{n}}|1_{\hat{n}}\rangle = 1, \quad (4.12)$$

and are mutually orthogonal

$$\langle 0_{\hat{n}}|1_{\hat{n}}\rangle = 0. \quad (4.13)$$

Note that we can always multiply eigenstates by an arbitrary phase factor so you might see expressions for these eigenstates which look different from Eqs. (4.11a) and (4.11b), but which are actually equivalent.

If we consider a direction $(\theta, \phi)$, then the eigenstate of spin in that direction with eigenvalue $+1$ is given by Eq. (4.11a). Similarly, the eigenstate with eigenvalue $-1$ is given by Eq. (4.11b), which is the antipodal point where $\theta \to \pi - \theta, \phi \to \phi + \pi$ see Fig. 4.1.

It is useful to consider three special cases of Eqs. (4.11):

- $(\theta = \phi = 0)$, the $z$ direction. Clearly $|0_{\hat{z}}\rangle = |0\rangle$ and $|1_{\hat{z}}\rangle = |1\rangle$ as required.

- $(\theta = \pi/2, \phi = 0)$, the $x$ direction:

$$|0_{\hat{x}}\rangle = \frac{1}{\sqrt{2}} (|0\rangle + |1\rangle) = |+\rangle, \quad (4.14)$$

$$|1_{\hat{x}}\rangle = \frac{1}{\sqrt{2}} (-|0\rangle + |1\rangle) = |--\rangle. \quad (4.15)$$

These are the eigenstates of $X$ as expected. ($|1_{\hat{z}}\rangle$ has the opposite sign to the conventionally defined state $|--\rangle$, but the overall sign of a state is of no importance.)
4.2. NO CLONING THEOREM

A classical bit, 0 or 1, can be copied, i.e. cloned. You just observe it and create another one. With qubits, however, it turns out to be not possible to clone an arbitrary state. This is called the “no-cloning theorem”. It imposes an important limitation on our ability to manipulate quantum states. We now give the simple derivation of this important result.

Consider the general qubit state

\[ |\psi\rangle = \alpha |0\rangle + \beta |1\rangle, \]

where \(|\alpha|^2 + |\beta|^2 = 1\). (4.18)

\footnote{Note that \(|0_n\rangle\) is specified by two parameters. This is the correct number to describe a general qubit state for the following reason. A qubit vector has two complex components making a total of four. However, one of these can be eliminated because the state must be normalized, and another can be eliminated because an overall phase is unimportant. This leaves two parameters necessary to describe a general qubit state.}

Figure 4.1: The Bloch sphere.

- \((\theta = \pi/2, \phi = \pi/2)\), the \(y\) direction:

\[
|0_y\rangle = \frac{1}{\sqrt{2}} (|0\rangle + i |1\rangle),
\]

\[
|1_y\rangle = \frac{1}{\sqrt{2}} (-|0\rangle + i |1\rangle).
\]

These are the eigenstates of \(Y\) as expected.

Even if the qubit is not an electron spin, Eqs. (4.11) provide a convenient parametrization of a general qubit basis. In particular, state \(|0_n\rangle\) is a convenient description\(\textsuperscript{1}\) of an arbitrary qubit state. It is an +1 eigenstate of \(\vec{\sigma} \cdot \hat{n}\), where \(n\) is in direction \((\theta, \phi)\) with \(\theta\) and \(\phi\) the polar and azimuthal angles of a point on a unit sphere (often called the Bloch sphere) see Fig. 4.1.
We can’t determine the state by measuring it because a measurement gives $|0\rangle$ with probability $|\alpha|^2$ and $|1\rangle$ with probability $|\beta|^2$, i.e. it destroys the superposition.

Can we clone the state without measuring it? If so, there must be a unitary operator $U$ which acts on $|\psi\rangle$ and an ancilla qubit, which is initialized to $|0\rangle$ say, and clones $|\psi\rangle$ as follows:

$$U|\psi\rangle |0\rangle = |\psi\rangle |\psi\rangle. \quad (4.19)$$

We shall see that no such operator can exist, because operators in quantum mechanics are linear.

Suppose that

$$U|\psi\rangle |0\rangle = |\psi\rangle |\psi\rangle, \quad (4.20)$$

$$U|\phi\rangle |0\rangle = |\phi\rangle |\phi\rangle. \quad (4.21)$$

Then, by linearity,

$$U (|\psi\rangle + |\phi\rangle ) |0\rangle = |\psi\rangle |\psi\rangle + |\phi\rangle |\phi\rangle. \quad (4.22)$$

However, this is not a clone of $|\psi\rangle + |\phi\rangle$ which would be

$$(|\psi\rangle + |\phi\rangle ) (|\psi\rangle + |\phi\rangle ) = |\psi\rangle |\psi\rangle + |\psi\rangle |\phi\rangle + |\phi\rangle |\psi\rangle + |\phi\rangle |\phi\rangle. \quad (4.23)$$

There is an inconsistency so a unitary operator $U$ for cloning does not exist.

The no-cloning theorem will be an important limitation when designing quantum algorithms.

### 4.3 Entanglement and Bell states

A striking aspect of quantum states of more than one qubit, which seems mysterious and plays a crucial role in quantum algorithms, is called “entanglement”. Here we will illustrate this concept for the simplest case of two qubits.

Let’s suppose that the first qubit is in state $|\psi_1\rangle = \alpha_1|0\rangle + \beta_1|1\rangle$ and the second qubit is in state $|\psi_2\rangle = \alpha_2|0\rangle + \beta_2|1\rangle$. The state of the two-qubit system is the direct product

$$|\psi_1\rangle \otimes |\psi_2\rangle = \begin{pmatrix} \alpha_1 \\ \beta_1 \end{pmatrix} \otimes \begin{pmatrix} \alpha_2 \\ \beta_2 \end{pmatrix} = \begin{pmatrix} \alpha_1 \alpha_2 \\ \alpha_1 \beta_2 \\ \beta_1 \alpha_2 \\ \beta_1 \beta_2 \end{pmatrix}. \quad (4.24)$$

This is called a product state.

However, a general qubit state is not a product state. It can be written as

$$|\phi\rangle_2 = c_0|00\rangle + c_1|01\rangle + c_2|10\rangle + c_3|11\rangle, \quad (4.25)$$

or equivalently as

$$|\phi\rangle_2 = c_0|00\rangle + c_1|01\rangle + c_2|10\rangle + c_3|11\rangle = \sum_{x=0}^{3} c_x|x\rangle_2, \quad (4.26)$$

where the notation $|x\rangle_2$ indicates that we have a state of two qubits and the states of the individual qubits are represented by the bits of the integer $x$.

The product state has

$$c_0 = \alpha_1 \alpha_2, \quad c_1 = \alpha_1 \beta_2, \quad c_2 = \beta_1 \alpha_2, \quad c_3 = \beta_1 \beta_2, \quad (4.27)$$

and so satisfies

$$c_0 c_3 = c_1 c_2. \quad (4.28)$$
This is the condition for a 2-qubit state to be a product state. States which do not have this property are said to be entangled.

The most-studied entangled states are so-called Bell states which involve two qubits. They are named in honor of the physicist whose inequalities (to be discussed later) demonstrated that the description of nature provided by quantum mechanics is fundamentally different from the classical description. The Bell states are defined by

\[
|β_{00}\rangle = \frac{1}{\sqrt{2}} (|00\rangle + |11\rangle),
\]

\[
|β_{01}\rangle = \frac{1}{\sqrt{2}} (|01\rangle + |10\rangle),
\]

\[
|β_{10}\rangle = \frac{1}{\sqrt{2}} (|00\rangle - |11\rangle),
\]

\[
|β_{11}\rangle = \frac{1}{\sqrt{2}} (|01\rangle - |10\rangle).
\]

These four equations can be combined as follows:

\[
|β_{xy}\rangle = \frac{1}{\sqrt{2}} (|0y\rangle + (-1)^x|1\overline{y}\rangle),
\]

where \(\overline{y}\) is the complement of \(y\), i.e. \(\overline{y} = 1 - y\). The Bell states are clearly entangled.

There are correlations between the qubits in the Bell states (quite generally between the qubits in entangled states). For example, if we consider \(|β_{00}\rangle\) and do a measurement on qubit 1, then a measurement of qubit 2 (if performed) would find the same result with 100% probability. We will discuss quantum correlation in entangled states in some detail in Chapter 7 when we investigate the Einstein-Podolsky-Rosen (EPR) claim that quantum mechanics is incomplete.

For the case of two qubits, Eq. (4.27) is a convenient way to test if a state is a product state or entangled. In a more general case where we have, say, \(n = n_A + n_B\) qubits, we may want to know whether a partition of the system into the two subsystems \(A\), with \(n_A\) qubits, and \(B\), with \(n_B\) qubits, gives a product state, i.e.

\[
|ψ\rangle_n = |ψ_A\rangle_{n_A} \otimes |ψ_B\rangle_{n_B},
\]

or whether the state is entangled with respect to this partition. In this case, with more than \(n = 2\) qubits, there is no simple expression analogous to Eq. (4.27) for the \(2^n\) coefficients \(c_x\), \((x = 0, 1, \ldots, 2^n - 1)\), which indicates a product state. Instead, a systematic way to investigate whether such a state is entangled or a product state is to use the density matrix, discussed in Chapter 5.

Appendix

4.A Angular Momentum Eigenstates

Physics students learn about quantum states which are eigenstates of angular momentum. This appendix relates Bell states to spin angular momentum eigenstates of two electrons. It is intended for physics students and is not essential reading for students of other disciplines.

The spin of an electron \(\vec{s}\) is given by

\[
\vec{s} = \frac{\hbar}{2} \vec{σ},
\]

where \(\hbar\) is Planck’s constant divided by \(2\pi\) and

\[
σ_x \equiv X = \begin{pmatrix} 0 & 1 \\ 1 & 0 \end{pmatrix}, \quad σ_y \equiv Y = \begin{pmatrix} 0 & -i \\ i & 0 \end{pmatrix}, \quad σ_z \equiv Z = \begin{pmatrix} 1 & 0 \\ 0 & -1 \end{pmatrix}.
\]
In general, spin angular momentum states, |S, m⟩, are specified by two quantum numbers S and m. The total spin quantum number S is defined by

\[ s_x^2 + s_y^2 + s_z^2 = \hbar^2 S(S + 1), \]  

so |S, m⟩ is an eigenvalue of (\(\hat{s}\))^2 with eigenvalue \(\hbar^2 S(S + 1)\). The quantum number m is defined such that |S, m⟩ is an also eigenstate of \(\hat{s}_z\) with eigenvalue \(\hbar m\), where m ranges from \(-S\) to \(S\) in integer steps (so there are 2S + 1 values of m for a given S). Thus the spin of an electron has \(S = 1/2\), and its two basis states are |\(S = 1/2, m = 1/2\)⟩ and |\(S = 1/2, m = -1/2\)⟩, which are often written as |↑⟩ and |↓⟩ respectively. The latter notation indicates that one thinks of these two states as spin “up” and spin “down”. By convention, the correspondence between the basis states of the electron spin in physics, |↑⟩ and |↓⟩, and the computational basis states in quantum computer science, |0⟩ and |1⟩, is taken to be\(^2\)

\[ |↑⟩ \equiv |0⟩, \quad |↓⟩ \equiv |1⟩. \]  

If we have two particles with total spin quantum numbers \(S_1\) and \(S_2\) then, as shown in textbooks on quantum mechanics [Gri05], the “vector rule” for addition of angular momentum states that the total spin quantum number of the combined system, \(S_{tot}\), takes integer values between \(S_1 + S_2\) and \(|S_1 - S_2|\). Thus, two electrons can have combined total spin quantum number \(S_{tot} = 1\) (for which there are 3 values of \(m_{tot}\), namely 1, 0 and \(-1\), and \(S_{tot} = 0\) (for which there is only one value of \(m_{tot}\), namely 0). These are called “triplet” and “singlet” states respectively. Note that the total number of states works out right since there are \(2^2 = 4\) states out of which 3 have \(S_{tot} = 1\) and 1 has \(S_{tot} = 0\), (i.e. \(2 \times 2 = 3 + 1\)).

It is also shown in the quantum mechanics textbooks that the states of two spin-1/2 particles with specified values of \(S_{tot}\) and \(m_{tot}\) are given by

\[
\begin{align*}
|S_{tot} = 1, m_{tot} = 1⟩ &= |↑↑⟩ & \equiv |00⟩, \quad \tag{4.35a} \\
|S_{tot} = 1, m_{tot} = 0⟩ &= \frac{1}{\sqrt{2}} \left( |↑↓⟩ + |↓↑⟩ \right) & \equiv \frac{1}{\sqrt{2}} \left( |01⟩ + |10⟩ \right), \quad \tag{4.35b} \\
|S_{tot} = 1, m_{tot} = -1⟩ &= |↓↓⟩ & \equiv |11⟩, \quad \tag{4.35c} \\
|S_{tot} = 0, m_{tot} = 0⟩ &= \frac{1}{\sqrt{2}} \left( |↑↓⟩ - |↓↑⟩ \right) & \equiv \frac{1}{\sqrt{2}} \left( |01⟩ - |10⟩ \right). \quad \tag{4.35d}
\end{align*}
\]

Eqs. (4.35a)–(4.35c) are the triplet states while Eq. (4.35d) is the singlet state.

Comparing with Eqs. (4.28) we see that

\[
\begin{align*}
|S_{tot} = 1, m_{tot} = 1⟩ &= \frac{1}{\sqrt{2}} \left( |00⟩ + |10⟩ \right), \quad \tag{4.36a} \\
|S_{tot} = 1, m_{tot} = 0⟩ &= |00⟩, \quad \tag{4.36b} \\
|S_{tot} = 1, m_{tot} = -1⟩ &= \frac{1}{\sqrt{2}} \left( |00⟩ - |10⟩ \right), \quad \tag{4.36c} \\
|S_{tot} = 0, m_{tot} = 0⟩ &= |00⟩, \quad \tag{4.36d}
\end{align*}
\]

Equations (4.36) connect Bell states and angular momentum states, while Eqs. (4.35) connect computational basis states and angular momentum states.

In this Chapter we have encountered three sets of states which can describe 2 qubits:

- the computational basis states |xy⟩,

\(^2\)Since |↑⟩ has \(\sigma_z = +1\) a physicist might think that the correspondence ought to be |↑⟩ ≡ |1⟩ rather than |↑⟩ ≡ |0⟩. The reason for the defining the correspondence as in Eq. (4.34) seems to be that if the states have different energy, then |0⟩ or |↑⟩ is usually taken to be the lower energy state, while |1⟩ or |↓⟩ is the excited state.
• the Bell states $|\beta_{xy}\rangle$, and

• the angular momentum states $|S_{\text{tot}}, m_{\text{tot}}\rangle$.

Each of these forms a basis set. In quantum computing we generally use computational basis states but sometimes the Bell basis will be useful. However, there does not seem to be a use for angular momentum basis states in quantum computing.
Chapter 5

The Density Matrix

5.1 Introduction

We will be interested in situations where a system is in contact with another, possibly much larger, system. Let’s call the system of interest subsystem $A$, and denote the other system by subsystem $B$. We use the word “subsystem” for $A$ and $B$, since we now consider them as the two parts of the combined $AB$ system. We want to describe the properties of subsystem $A$ without explicitly including the degrees of freedom of subsystem $B$. This is accomplished by the “density matrix”. Two situations where the density matrix is useful are:

- To determine whether a state is a product state or entangled with respect to a partition of the system into two subsystems.

- Quantum computers, where $A$ is (some of) the qubits of the computer and $B$ is the environment which inevitably couples to the computational qubits. The environment is very complicated with a huge (essentially infinite) number of degrees of freedom, so some approximations will have to be made to determine the density matrix for $A$, the qubits of interest. We will discuss the effects of the environment on a quantum computer in the Chapter 19.

For further reading on the density matrix see Refs. [NC00, VAT16, RP14].

5.2 Definition of the Density Matrix

In this chapter we will assume that the combined $AB$ system is in a well-defined quantum state $|\psi_{AB}\rangle$. We write $|\psi_{AB}\rangle$ as a superposition as follows:

$$|\psi_{AB}\rangle = \sum_{i=1}^{N_A} \sum_{n=1}^{N_B} c_{in} |i_A n_B\rangle,$$

(5.1)

where $N_A (= 2^{n_A})$ is the number of states of subsystem $A$, $N_B (= 2^{n_B})$ is the number of states of subsystem $B$, the $|i_A\rangle$ are a basis for $A$, the $|n_B\rangle$ are a basis for $B$, and the $c_{in}$ are amplitudes. Where necessary, for clarity we put a subscript on the basis state labels, e.g. $i_A$ rather than just $i$, and $n_B$ rather than just $n$, to indicate which subsystem the label refers to. Because $|\psi_{AB}\rangle$ is normalized we have

$$\sum_{i,n} |c_{in}|^2 = 1.$$ 

(5.2)
We want to compute the expectation value of some operator $\hat{O}_A$ acting only on the $A$ degrees of freedom, i.e.

$$\langle \hat{O}_A \rangle = \langle \psi_{AB} | \hat{O}_A | \psi_{AB} \rangle = \sum_{i,i'} \sum_{n,n'} \langle n'| \hat{O}_A | n \rangle \langle i' | i \rangle c_{i'n'}^* c_{in}$$

where in the third line we used that $\hat{O}_A$ does not depend on the $B$ degrees of freedom, and in the fourth line we used that $\langle n'| n \rangle = \delta_{nn'}$.

We define the density matrix for subsystem $A$ by

$$\langle i | \rho^A | i' \rangle = \sum_n c_{in} c_{i'n}^* ,$$

omitting for conciseness the label $A$ on $|i_A\rangle$ and $|i'_A\rangle$ when there is no ambiguity. Hence

$$\langle \hat{O}_A \rangle = \sum_{i,i'} \langle i | \rho^A | i' \rangle \langle i' | \hat{O}_A | i \rangle = \sum_i \langle i | \rho^A \hat{O}_A | i \rangle = \text{Tr}_A \left( \rho^A \hat{O}_A \right),$$

where Tr $(\cdots)$ means the trace (sum of diagonal elements), and we used that $\sum_{i'} |i'\rangle \langle i'| = 1$, the completeness relation (if this is not obvious, think about the summation index in the expression for matrix multiplication).

Note:

- In the definition for $\rho^A$ in Eq. (5.4) we see that the two labels for the states of subsystem $B$ ($n$ and $n'$) are set equal and summed over. We therefore say that the effects of subsystem $B$ have been “traced out” or “traced over”.

- One can equivalently trace out the degrees of freedom in $A$ to get the density matrix for subsystem $B$, i.e.

$$\langle n | \rho^B | n' \rangle = \sum_i c_{in} c_{in'}^* .$$

- We see that

$$\text{Tr} \rho_A = \sum_i \langle i | \rho^A | i \rangle = \sum_j c_{in} c_{in}^* = 1,$$

where we used Eq. (5.2). Hence the trace of a density matrix is one.

\(^1\)More elegantly, one can write Eq. (5.4) as $\rho^A = \sum_n \langle n_B | \psi_{AB} \rangle \langle \psi_{AB} | n_B \rangle = \text{Tr}_B | \psi_{AB} \rangle | \psi_{AB} \rangle$. As stated above, in this chapter we assume that the combined $AB$ system is in single quantum state. If, instead, the combined system is itself described by a density matrix $\rho^{AB}$, then the reduced density matrix for subsystem $A$ is given by $\rho^A = \text{Tr}_B \rho^{AB}$.
5.3. Determining if a state is entangled

As we shall see, it is useful to diagonalize the density matrix, obtaining its eigenvalues $\lambda_\alpha$ and eigenvectors $|\phi_\alpha\rangle$. Since the sum of the eigenvalues is equal to the trace we have, according to Eq. (5.7),

$$\sum_\alpha \lambda_\alpha = 1,$$

which suggests that the eigenvalues can be interpreted as probabilities (since probabilities also sum to 1). We shall now see that this interpretation is correct.

Let’s consider Eq. (5.5) in the basis where $\rho^A$ is diagonal. We have

$$\langle \hat{O}_A \rangle = \text{Tr} \left( \rho^A \hat{O}_A \right) = \sum_\alpha \lambda_\alpha \langle \phi_\alpha | \hat{O}_A | \phi_\alpha \rangle.$$  

(5.9)

Thus we get the expectation value of $\hat{O}_A$ in state $|\psi_{AB}\rangle$ by (i) computing the expectation of $\hat{O}_A$ in state $|\phi_\alpha\rangle$ (an eigenvector of $\rho^A$), (ii) multiplying by $\lambda_\alpha$ (the corresponding eigenvalue of $\rho^A$), and (iii) summing over $\alpha$. This clearly shows that $\lambda_\alpha$ should be thought of as the probability that subsystem $A$ is in state $|\alpha\rangle$. To emphasize this, from now on we will denote the eigenvalues of the density matrix by $p_\alpha$.

It is important to note that the density matrix $\rho^A$ is the same no matter what quantity of system $A$ is to be calculated. Thus $\rho^A$ only has to be calculated once.

5.3 Determining if a state is entangled

One use of the density matrix is that it gives a systematic prescription for determining whether a state is a product state or entangled with respect to a partition into subsystems $A$ and $B$. If it is not a product state we say that it is a mixed state and is “entangled” with respect to this partition.

To see how the density matrix can determine if a state is a product state or is entangled with respect to partition into $A$-$B$ subsystems, let’s assume initially that $|\psi_{AB}\rangle$ is a product state, i.e.

$$|\psi_{AB}\rangle = |\phi\rangle_A |\mu\rangle_B.$$  

(5.10)

In this case subsystem $A$ is definitely in state $|\phi\rangle$, so the eigenvalues of $\rho^A$ must be $p_1 = 1$ and $p_\alpha = 0$ for $\alpha \neq 1$. Also the eigenvector for the first eigenvalue must be given by $|\phi_1\rangle = |\phi\rangle$.

Hence, if the state of the combined $AB$ system is a product state then one of the eigenvalues of the density matrix of $A$ (or of $B$) will be 1 and the others zero. Conversely if more than one of the eigenvalues of the density matrix are positive (since they are probabilities they can only be positive or zero) the state is mixed, i.e. entangled.

It is actually not necessary to diagonalize the density matrix to determine if the state is a product state or entangled. Instead it is sufficient to take its square. To see this note that

$$\text{Tr} \left( \rho^A \right)^2 = \sum_\alpha p_\alpha^2.$$  

(5.11)

Since the $p_\alpha$ must lie between 0 and 1 and $\sum_\alpha p_\alpha = 1$, one can show that $\sum_\alpha p_\alpha^2 \leq 1$, with the equality only holding if one of the $p_\alpha$ is 1 and the others zero. As an example, consider the case of two states, for which the eigenvalues are $p$ and $1-p$ with $0 \leq p \leq 1$. Now

$$\text{Tr} \left( \rho^A \right)^2 = \sum_{\alpha=1}^2 p_\alpha^2 = p^2 + (1-p)^2 = 1 - 2p + 2p^2 = 1 - 2p(1-p).$$  

(5.12)
For $0 \leq p \leq 1$, we see that $0 \leq 2p(1-p) \leq 1/2$ and is only zero for $p = 0$ and 1. Consequently, \( \text{Tr} \left( \rho^A \right)^2 < 1 \) unless $p = 0$ or 1.

Hence we have the following general criterion:

\[
\text{if } \text{Tr} \left( \rho^A \right)^2 = 1, \quad \text{then we have a product state},
\]
\[
\text{if } \text{Tr} \left( \rho^A \right)^2 < 1, \quad \text{then we have a mixed (entangled) state},
\]

(5.13)

We emphasize again that \( \text{Tr} \rho^A = 1 \) always.

Sometimes one defines the Von Neumann entanglement entropy by

\[
S(\rho^A) = -\text{Tr} \rho^A \log \rho^A = -\sum_\alpha p_\alpha \log p_\alpha.
\]

(5.14)

It is easy to see that \( S(\rho^A) = 0 \) if the state is a product state. In the opposite limit, of a uniformly mixed state where $p_\alpha = 1/N_A$ for all $\alpha$, one has \( S(\rho^A) = \log N_A \). For the case where subsystem $A$ is a single qubit, this gives \( S(\rho^A) = \log 2 \).

### 5.4 Some Simple Examples

In this section we consider some simple examples where subsystems $A$ and $B$ each have just a single qubit.

#### 5.4.1 Example 1:

We take

\[
|\psi_{AB}\rangle = \frac{1}{2} (|0_A0_B\rangle + |0_A1_B\rangle - |1_A0_B\rangle - |1_A1_B\rangle)
\]

(5.15)

**Note:** We can see “by inspection” that this is a product state

\[
|\psi_{AB}\rangle = \frac{1}{\sqrt{2}} (|0_A\rangle - |1_A\rangle) \otimes \frac{1}{\sqrt{2}} (|0_B\rangle + |1_B\rangle).
\]

(5.16)

We shall now show how this result is obtained from the density matrices $\rho^A$ and $\rho^B$.

We have

\[
c_{00} = c_{01} = \frac{1}{2}, \quad c_{10} = c_{11} = -\frac{1}{2},
\]

(5.17)

so

\[
\rho^A_{00} = c_{00}c_{00} + c_{01}c_{01} = \frac{1}{2}
\]

\[
\rho^A_{01} = c_{00}c_{10} + c_{01}c_{11} = -\frac{1}{2}
\]

\[
\rho^A_{10} = c_{10}c_{00} + c_{11}c_{01} = -\frac{1}{2}
\]

\[
\rho^A_{11} = c_{10}c_{10} + c_{11}c_{11} = \frac{1}{2},
\]

(5.18)

and hence

\[
\rho^A = \frac{1}{2} \begin{pmatrix} 1 & -1 \\ -1 & 1 \end{pmatrix}.
\]

(5.19)

The eigenvalues are given by

\[
\begin{vmatrix} \frac{1}{2} - \lambda & -\frac{1}{2} \\ -\frac{1}{2} & \frac{1}{2} - \lambda \end{vmatrix} = 0
\]

(5.20)

so

\[
(\lambda - \frac{1}{2})^2 - \left(-\frac{1}{2}\right)^2 = 0
\]

(5.21)
which gives $\lambda = 1$ and 0. Since only one eigenvalue is non-zero this is a separable state, as we saw above.

The eigenvector with eigenvalue $\lambda = 1$ is given by
\[
\frac{1}{2} \begin{pmatrix} 1 & 1 \\ -1 & 1 \end{pmatrix} \begin{pmatrix} a \\ b \end{pmatrix} = \begin{pmatrix} a \\ b \end{pmatrix}.
\]
(5.22)

Both the resulting equations give $b = -a$ so the normalized eigenvector is
\[
|\phi_{1,A}\rangle = \frac{1}{\sqrt{2}} \left( |0\rangle_A - |1\rangle_A \right).
\]
(5.23)

Hence, with probability 1, subsystem $A$ is in state $|\phi_1\rangle$, in agreement with Eq. (5.16).

One can repeat the same calculation for $\rho^B$. The results are
\[
\rho_{00}^B = c_{00}c_{00} + c_{10}c_{10} = \frac{1}{2},
\rho_{01}^B = c_{00}c_{01} + c_{10}c_{11} = \frac{1}{2},
\rho_{10}^B = c_{01}c_{00} + c_{11}c_{10} = \frac{1}{2},
\rho_{11}^B = c_{01}c_{01} + c_{11}c_{11} = \frac{1}{2},
\]
so
\[
\rho^B = \frac{1}{2} \begin{pmatrix} 1 & 1 \\ 1 & 1 \end{pmatrix}.
\]
(5.25)

The eigenvalues are given by
\[
\begin{vmatrix} \frac{1}{2} - \lambda & \frac{1}{2} \\ \frac{1}{2} & \frac{1}{2} - \lambda \end{vmatrix} = 0
\]
(5.26)

so
\[
\left( \lambda - \frac{1}{2} \right)^2 - \left( \frac{1}{2} \right)^2 = 0
\]
(5.27)

which gives $\lambda = 1$ and 0, the same as for $\rho^A$. It is true in general that the non-zero eigenvalues of $\rho^A$ and $\rho^B$ must be equal, provided that the combined $AB$ system is in a single quantum state. This is discussed further in the more advanced material in Appendix 5.A

The eigenvector with eigenvalue $\lambda = 1$ is given by
\[
\frac{1}{2} \begin{pmatrix} 1 & 1 \\ 1 & 1 \end{pmatrix} \begin{pmatrix} a \\ b \end{pmatrix} = \begin{pmatrix} a \\ b \end{pmatrix}.
\]
(5.28)

Both the resulting equations give $b = a$ so the normalized eigenvector is
\[
|\sigma_{1,B}\rangle = \frac{1}{\sqrt{2}} \left( |0\rangle_B + |1\rangle_B \right).
\]
(5.29)

Hence, with probability 1, subsystem $B$ is in state $|\sigma_1\rangle$, again in agreement with Eq. (5.16).

### 5.4.2 Example 2:

In this example we take one of the Bell states,
\[
|\psi_{AB}\rangle = \frac{1}{\sqrt{2}} \left( |0\rangle_A |0\rangle_B + |1\rangle_A |1\rangle_B \right),
\]
(5.30)

which is clearly entangled. Here we have
\[
c_{00} = c_{11} = \frac{1}{\sqrt{2}}, \quad c_{10} = c_{01} = 0.
\]
(5.31)
CHAPTER 5. THE DENSITY MATRIX

Hence

\[ \rho_A^{00} = c_{00}c_{00} + c_{01}c_{01} = \frac{1}{2} \]
\[ \rho_A^{01} = c_{00}c_{10} + c_{01}c_{11} = 0 \]  
\[ \rho_A^{10} = c_{10}c_{00} + c_{11}c_{01} = 0 \]
\[ \rho_A^{11} = c_{10}c_{10} + c_{11}c_{11} = \frac{1}{2}, \]  

so

\[ \rho_A = \frac{1}{2} \begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix}. \]  

(5.33)

This is already in diagonal form so we read off that the two eigenvalues are both equal to 1/2. Since more than one eigenvalue is positive, the state is entangled. A density matrix like this, with all eigenvalues equal, is said to be maximally entangled.

It is easy to see that the same eigenvalues are obtained from \( \rho_B \).

5.4.3 Example 3:

This example is slightly more complicated but it is useful to go through it in detail. We take

\[ |\psi_{AB}\rangle = \frac{1}{\sqrt{8}} \left( |0_A0_B\rangle + \sqrt{3}|0_A1_B\rangle - \sqrt{3}|1_A0_B\rangle - |1_A1_B\rangle \right). \]  

(5.34)

so

\[ c_{00} = \frac{1}{\sqrt{8}}, \quad c_{01} = \frac{3}{8}, \quad c_{10} = -\frac{3}{8}, \quad c_{11} = -\frac{1}{\sqrt{8}}. \]  

(5.35)

It follows that

\[ \rho_A^{00} = c_{00}c_{00} + c_{01}c_{01} = \frac{1}{2} \]
\[ \rho_A^{01} = c_{00}c_{10} + c_{01}c_{11} = -\frac{\sqrt{3}}{4} \]
\[ \rho_A^{10} = c_{10}c_{00} + c_{11}c_{01} = -\frac{\sqrt{3}}{4} \]
\[ \rho_A^{11} = c_{10}c_{10} + c_{11}c_{11} = \frac{1}{2}, \]  

so

\[ \rho_A = \frac{1}{4} \begin{pmatrix} 2 & -\sqrt{3} \\ -\sqrt{3} & 2 \end{pmatrix}. \]  

(5.37)

The eigenvalues are found to be

\[ p_1 = \frac{1}{4} \left( 2 + \sqrt{3} \right), \quad p_2 = \frac{1}{4} \left( 2 - \sqrt{3} \right), \]  

(5.38)

with corresponding eigenvectors

\[ |\phi_{1,A}\rangle = \frac{1}{\sqrt{2}} (|0_A\rangle - |1_A\rangle) \]
\[ |\phi_{2,A}\rangle = \frac{1}{\sqrt{2}} (|0_A\rangle + |1_A\rangle). \]  

(5.39)

Thus subsystem \( A \) can be regarded as being in state \( |\phi_1\rangle \) with probability \( p_1 \) and in state \( |\phi_2\rangle \) with probability \( p_2 \).

Repeating the same arguments for \( \rho_B \) gives

\[ \rho_B^{00} = c_{00}c_{00} + c_{10}c_{10} = \frac{1}{2} \]
\[ \rho_B^{01} = c_{00}c_{01} + c_{10}c_{11} = \frac{\sqrt{3}}{4} \]
\[ \rho_B^{10} = c_{01}c_{00} + c_{11}c_{10} = \frac{\sqrt{3}}{4} \]
\[ \rho_B^{11} = c_{01}c_{01} + c_{11}c_{11} = \frac{1}{2}, \]  

(5.40)
so
\[
\rho^B = \frac{1}{4} \begin{pmatrix} 2 & \sqrt{3} \\ \sqrt{3} & 2 \end{pmatrix}.
\] (5.41)

The eigenvalues are found to be again given by Eq. (5.38) and the corresponding eigenvectors are
\[
|\sigma_{1,B}\rangle = \frac{1}{\sqrt{2}} (|0_B\rangle + |1_B\rangle)
\]
\[
|\sigma_{2,B}\rangle = \frac{1}{\sqrt{2}} (-|0_B\rangle + |1_B\rangle).
\] (5.42)

Subsystem \(B\) can therefore be regarded as being in state \(|\sigma_1\rangle\) with probability \(p_1\) and in state \(|\sigma_2\rangle\) with probability \(p_2\).

It is interesting to note that if we define
\[
c_1 = \frac{1}{2} \sqrt{2 + \sqrt{3}}, \quad c_2 = \frac{1}{2} \sqrt{2 - \sqrt{3}},
\] (5.43)

so
\[
p_1 = c_1^2, \quad p_2 = c_2^2,
\] (5.44)

then a bit of algebra\(^2\) shows that
\[
|\psi_{AB}\rangle = c_1|\phi_{1,A}\rangle \otimes |\sigma_{1,B}\rangle + c_2|\phi_{2,A}\rangle \otimes |\sigma_{2,B}\rangle.
\] (5.45)

This is an example of Schmidt decomposition which is described in the more advanced material in Appendix 5.A. The coefficients \(c_1\) and \(c_2\) are known as Schmidt coefficients.

According to Eq. (5.45) we can decompose \(|\psi_{AB}\rangle\) in the following way: with probability \(p_1 = c_1^2\) subsystem \(A\) is in state \(|\psi_{1,A}\rangle\) and subsystem \(B\) is in state \(|\sigma_{1,B}\rangle\), and with probability \(p_2 = c_2^2 (= 1-p_1)\) subsystem \(A\) is in state \(|\psi_{2,A}\rangle\) and subsystem \(B\) is in state \(|\sigma_{2,B}\rangle\). In this way one can see why the non-zero eigenvalues of the two subsystem density matrices \(\rho^A\) and \(\rho^B\) must be equal, namely for both matrices the eigenvalues are given by \(c_1^2\) and \(c_2^2\).

5.5 Conclusions

We have seen that the density matrix is useful when studying the properties of a system composed of two subsystems \(A\) and \(B\). More precisely, it can be used to:

- Determine the properties of one of the subsystems \(A\) without explicitly having to include the degrees of freedom of the other subsystem \(B\). This is particularly useful if \(B\) contains a very large number of degrees of freedom. An example of a large “subsystem” is the environment, with which, unfortunately, the qubits of a quantum computer unavoidably interact.

- If the combined \(AB\) system is in a single state, the properties of the subsystem density matrices tell us whether that state is a product state with respect to the \(A-B\) partition or whether, on the other hand, it is a mixed state in which the two subsystems are entangled.

\(^2\)Note that \(\sqrt{2 + \sqrt{3}} - \sqrt{2 - \sqrt{3}} = \sqrt{2}\) and \(\sqrt{2 + \sqrt{3}} + \sqrt{2 - \sqrt{3}} = \sqrt{6}\), which are proved by squaring both sides.
Appendices

5.A Schmidt Decomposition

(This is more advanced material which is not required for the course.)

It can be shown \([NC00]\) that a state \(|\psi_{AB}\rangle\) can be written as

\[
|\psi_{AB}\rangle = \sum_{\alpha} c_{\alpha} |\phi_{\alpha,A}\rangle \otimes |\sigma_{\alpha,B}\rangle,
\]

(5.46)

where the number of terms is less than or equal to the smaller of \(N_A\) and \(N_B\), and the \(|\phi_{\alpha}\rangle\) are mutually orthogonal as are the \(|\sigma_{\alpha}\rangle\). This is known as the Schmidt decomposition and the \(c_{\alpha}\) are called Schmidt coefficients. Since the phases of \(|\phi_{\alpha,A}\rangle\) and \(|\sigma_{\alpha,B}\rangle\) can be chosen independently, one can always choose these to make the \(c_{\alpha}\) real and non-negative. Using the more elegant definition of \(\rho_A\) given in footnote on page 40 and working in the \(|\phi_{\alpha}\rangle\) basis for the states of \(A\) and the \(|\sigma_{\alpha}\rangle\) basis for the states of \(B\), one has:

\[
\rho^A = \text{Tr}_B |\psi_{AB}\rangle\langle\psi_{AB}| = \sum_{\alpha} c_{\alpha}^2 |\phi_{\alpha,A}\rangle \langle\phi_{\alpha,A}|.
\]

(5.47)

This shows that \(\rho^A\) has non-zero eigenvalues \(p_{\alpha} = c_{\alpha}^2\) with corresponding eigenvectors \(|\phi_{\alpha}\rangle\). Similarly one has:

\[
\rho^B = \text{Tr}_A |\psi_{AB}\rangle\langle\psi_{AB}| = \sum_{\alpha} c_{\alpha}^2 |\sigma_{\alpha,B}\rangle \langle\sigma_{\alpha,B}|,
\]

(5.48)

which shows that \(\rho^B\) has non-zero eigenvalues \(p_{\alpha} = c_{\alpha}^2\) (the same as for \(\rho^A\)) with corresponding eigenvectors \(|\sigma_{\alpha}\rangle\). The number of non-zero Schmidt coefficients (the \(c_{\alpha}\)) is called the Schmidt number (or Schmidt rank). If the Schmidt number is 1 the state is a product state, while if it is greater than 1, the state is entangled.

5.B Change in the density matrix under a unitary transformation

If qubit \(A\) (more generally subsystem \(A\)) is acted by a unitary transformation \(U^A\) then we show now that the density matrix for subsystem \(A\) changes from \(\rho^A\) to \(\rho'^A\) where:

\[
\rho'^A = U^A \rho^A (U^A)^\dagger.
\]

(5.49)

To see this, note that \(|\psi_{AB}\rangle\) in Eq. (5.1) goes to \(|\psi'_{AB}\rangle\) where

\[
|\psi'_{AB}\rangle = \sum_{i,j} c'_{ij} |i_A\rangle \otimes |j_B\rangle
\]

(5.50)

in which

\[
c'_{ij} = \sum_k U^A_{ik} c_{kj}
\]

(5.51)

\[3\] We are using here the outer product notation for representing matrices, which is briefly described in Sec. 3.4.
describes the change in amplitudes produced by the action of $U^A$. Note that the second index $j$ on the amplitude $c_{ij}$ refers to subsystem $B$ and is not changed. Hence

$$
\rho^{A\prime}_{i,i'} = \sum_j c^{A\prime}_{ij} c^{A\prime*}_{i'j}
$$

$$
= \sum_{j,k_1,k_2} U^A_{ik_1} c_{k_1j} U^{A*}_{i'k_2} c^{*}_{k_2j}
$$

$$
= \sum_{k_1,k_2} U^A_{ik_1} \left( \sum_j c_{k_1j} c^{*}_{k_2j} \right) U^{A*}_{i'k_2}
$$

$$
= \sum_{k_1,k_2} U^A_{ik_1} \rho^A_{k_1,k_2} (U^A_{k_2'i'})\dagger
$$

$$
= \left( U^A \rho^A (U^A)^\dagger \right)_{i,i'},
$$

so we obtain Eq. (5.49).

Note that the most general operation that can be applied to the combined $AB$ system is a unitary transformation acting on the whole system, not just on subsystem $A$. One can show that if one performs such a general unitary operation on the combined system, and then recomputes the density matrix of subsystem $A$, the new density matrix is not in general related to the old one by a unitary transformation. This is how irreversible processes can occur in a subsystem which is coupled to the environment. A more detailed discussion of this is beyond the scope of the course but the interested student is referred to Nielsen and Chuang [NC00] and Rieffel and Polak [RP14].
Chapter 6

Coherent Superposition Versus Incoherent Addition of Probabilities

The purpose of this chapter is to clarify the distinction between a coherent superposition of amplitudes in quantum mechanics and an incoherent (classical) addition of probabilities.

6.1 Coherent Linear Superposition: 1 qubit

To illustrate coherent superposition, consider one qubit in the following state

$$|\psi\rangle = \alpha|0\rangle + \beta|1\rangle,$$

(6.1)

where $|\alpha|^2 + |\beta|^2 = 1$. We denote $|\alpha|^2$ by $p$. Evidently $|\psi\rangle$ is a linear superposition of basis states $|0\rangle$ and $|1\rangle$. We say it is a coherent superposition because there is a well defined phase relationship between the pieces in the superposition, which means that there can be interference between these pieces in subsequent operations.

If we measure $|\psi\rangle$ in the computational basis we get

- $|0\rangle$ with probability $|\alpha|^2 = p$,
- $|1\rangle$ with probability $|\beta|^2 = 1 - p$.

To show the effects of interference we apply a Hadamard gate, defined in Eq. (2.27), before doing the measurement. The result is

$$|\psi\rangle = H|\psi\rangle = \frac{\alpha}{\sqrt{2}}(|0\rangle + |1\rangle) + \frac{\beta}{\sqrt{2}}(|0\rangle - |1\rangle) = \left(\frac{\alpha + \beta}{\sqrt{2}}\right)|0\rangle + \left(\frac{\alpha - \beta}{\sqrt{2}}\right)|1\rangle.$$

(6.3)

If we do a measurement in the computational basis after applying the Hadamard, the results are

- $|0\rangle$ with probability $\frac{1}{2}|\alpha + \beta|^2 = \frac{1}{2}(1 + \alpha\beta^* + \alpha^*\beta)$,
- $|1\rangle$ with probability $\frac{1}{2}|\alpha - \beta|^2 = \frac{1}{2}(1 - \alpha\beta^* - \alpha^*\beta)$.

(6.4)

The factor $\alpha\beta^* + \alpha^*\beta$ comes from interference between the two pieces in the linear combination of $|\psi\rangle$ in Eq. (6.1). In particular, if $\alpha = \beta = \frac{1}{\sqrt{2}}$, so $p = \frac{1}{2}$, we get

- $|0\rangle$ with probability 1,
- $|1\rangle$ with probability 0,

(6.5)

showing that there is zero probability of getting state $|1\rangle$ for $\alpha = \beta = \frac{1}{\sqrt{2}}$ if we measure after performing a Hadamard. The vanishing probability of getting $|1\rangle$ is due to destructive interference between the two pieces of the superposition in state $|\psi\rangle$ in Eq. (6.1).
6.2 Incoherent (Classical) Addition of Probabilities: 2 qubits

To illustrate the incoherent addition of probabilities consider two qubits in the following entangled state

$$|\psi_2\rangle = \alpha|00\rangle + \beta|11\rangle,$$

(6.6)

where we again denote $|\alpha|^2$ by $p$. If $\alpha = \pm \beta = \frac{1}{\sqrt{2}}$ this is a Bell state. Let us write $|\psi_2\rangle$ more explicitly as

$$|\psi_2\rangle = \alpha|0_A\rangle \otimes |0_B\rangle + \beta|1_A\rangle \otimes |1_B\rangle.$$  

(6.7)

If we focus on qubit $A$, say, then state $|\psi_2\rangle$ looks rather similar to the 1-qubit state $|\psi\rangle$ in Eq. (6.1), in that there is a piece where qubit $A$ is $|0\rangle$ with amplitude $\alpha$ and a piece where qubit $A$ is $|1\rangle$ with amplitude $\beta$. However, for $|\psi_2\rangle$, unlike for $|\psi\rangle$, each of these pieces goes with a different state for qubit $B$ (i.e. $|\psi_2\rangle$ is entangled). Because of this entanglement, we do net get interference between the pieces of $|\psi_2\rangle$ if we perform operations on qubit $A$ followed by a measurement of that qubit, as we shall now show.

We focus on the behavior of qubit $A$ by computing its density matrix, see Chapter 5. Writing

$$|\psi_2\rangle = \sum_{i,j=0}^1 c_{ij}|i_A\rangle \otimes |j_B\rangle$$

(6.8)

we have here $c_{00} = \alpha, c_{11} = \beta, c_{01} = c_{10} = 0$. As shown in Chapter 5 the elements of the density matrix for $A$ are given by

$$\rho_{i,j}^A = \sum_{j=0}^1 c_{ij}^* c_{ij}^j$$

(6.9)

so here $\rho_{00}^A = c_{00}^* c_{00} = |\alpha|^2 = p, \rho_{11}^A = c_{11}^* c_{11}^1 = |\beta|^2 = 1 - p$ and $\rho_{01}^A = \rho_{10}^A = 0$. Thus we have

$$\rho^A = \begin{pmatrix} p & 0 \\ 0 & 1 - p \end{pmatrix}.$$  

(6.10)

Trivially, the eigenvalues are $p$ and $1 - p$ with corresponding eigenvectors $|0\rangle$ and $|1\rangle$. As discussed in Chapter 5, this means that if we focus on qubit $A$, performing unitary operations and measurements just on this qubit, then the qubit can be regarded as initially being in state $|0\rangle$ with probability $p$ and in state $|1\rangle$ with probability $1 - p$.

If we measure qubit $A$ before doing any operation on it we get

- $|0\rangle$ with probability $p$,
- $|1\rangle$ with probability $1 - p$,

(6.11)

which is the same as in Eq. (6.2) for a single qubit in a coherent superposition.

However, a difference appears if we perform a unitary transformation on qubit $A$ before measuring it. Here we apply a Hadamard as we did in Sec. 6.1. Before acting with $H$, the state of qubit $A$ is described by the density matrix in Eq. (6.10), which means, as stated above, that qubit $A$ is in state $|0\rangle$ with probability $p$ and in state $|1\rangle$ with probability $1 - p$. Hence, after acting with $H$, qubit $A$ is\footnote{We will verify this in Appendix 6.A by directly computing the new density matrix after $H$ has acted on qubit $A$.}

in state $H|0\rangle = \frac{1}{\sqrt{2}} (|0\rangle + |1\rangle)$ with probability $p$,

in state $H|1\rangle = \frac{1}{\sqrt{2}} (|0\rangle - |1\rangle)$ with probability $1 - p$.

(6.12)

We then measure in the computational basis. If qubit $A$ is in state $H|0\rangle$ (which occurs with probability $p$) one has probability $\frac{1}{2}$ to get $|0\rangle$ and probability $\frac{1}{2}$ to get $|1\rangle$. If qubit $A$ is in state $H|1\rangle$ (which occurs
with probability $1 - p$) one again has probability $\frac{1}{2}$ to get $|0\rangle$ and probability $\frac{1}{2}$ to get $|1\rangle$. Combining these possible outcomes, one obtains from a measurement of qubit $A$:

$$
|0\rangle \text{ with probability } \frac{1}{2}(p + 1 - p) = \frac{1}{2},
$$

$$
|1\rangle \text{ with probability } \frac{1}{2}(p + 1 - p) = \frac{1}{2},
$$

(6.13)

which is independent of $p$. Equation (6.13) differs from Eq. (6.4), the case of a coherent superposition, by the absence of the factors of $\alpha\beta^* + \alpha^*\beta$ which came from interference.

To summarize, for a coherent superposition one sums the amplitudes and then squares, e.g.

$$
\frac{1}{2} |\alpha + \beta|^2,
$$

(6.14)

while for an incoherent superposition one squares and then sums, e.g.

$$
\frac{1}{2} (|\alpha|^2 + |\beta|^2).
$$

(6.15)

**Appendices**

**6.A Computation of the Density Matrix $\rho^A$ after the action of the Hadamard**

In this appendix we verify that Eq. (6.12) is correct by working out from scratch the density matrix for the state

$$
|\psi'_2\rangle = H_A|\psi_2\rangle
$$

$$
= \alpha (H_A|0_A\rangle) \otimes |0_B\rangle + \beta (H_A|1_A\rangle) \otimes |1_B\rangle
$$

(6.16)

$$
= \frac{\alpha}{\sqrt{2}} (|0_A0_B\rangle + |1_A0_B\rangle) + \frac{\beta}{\sqrt{2}} (|0_A1_B\rangle - |1_A1_B\rangle).
$$

The coefficients are

$$
c_{00} = \frac{\alpha}{\sqrt{2}}, \quad c_{10} = \frac{\alpha}{\sqrt{2}}, \quad c_{01} = \frac{\beta}{\sqrt{2}}, \quad c_{11} = -\frac{\beta}{\sqrt{2}}.
$$

(6.17)

so the elements of the density matrix $\rho^A$ are given by

$$
\rho_{00}^A = c_{00}c_{00} + c_{01}c_{01} = \frac{1}{2} (|\alpha|^2 + |\beta|^2) = \frac{1}{2}
$$

$$
\rho_{01}^A = c_{00}c_{10} + c_{01}c_{11} = \frac{1}{2} (|\alpha|^2 - |\beta|^2) = \frac{1}{2}(2p - 1)
$$

$$
\rho_{10}^A = c_{10}c_{00} + c_{11}c_{01} = \frac{1}{2} (|\alpha|^2 - |\beta|^2) = \frac{1}{2}(2p - 1)
$$

$$
\rho_{11}^A = c_{10}c_{10} + c_{11}c_{11} = \frac{1}{2} (|\alpha|^2 + |\beta|^2) = \frac{1}{2}
$$

(6.18)

and hence

$$
\rho^A = \frac{1}{2} \begin{pmatrix}
1 & 2p - 1 \\
2p - 1 & 1
\end{pmatrix}.
$$

(6.19)

The eigenvalues are given by

$$
\begin{vmatrix}
\frac{1}{2} - \lambda & p - \frac{1}{2} \\
\frac{1}{2} - \lambda & \frac{1}{2} - \lambda
\end{vmatrix}
$$

(6.20)

which gives

$$
(\lambda - \frac{1}{2})^2 - (p - \frac{1}{2})^2 = 0.
$$

(6.21)
which can be written as
\[
\lambda^2 - \lambda + p - p^2 = (\lambda - p)(\lambda - 1 + p) = 0,
\] (6.22)
so the solutions are
\[
\lambda = p \text{ and } 1 - p.
\] (6.23)

For \(\lambda = p\) one finds that the eigenvector is \(\frac{1}{\sqrt{2}}(|0\rangle + |1\rangle)\), while for \(\lambda = 1 - p\) one finds that the eigenvector is \(\frac{1}{\sqrt{2}}(|0\rangle - |1\rangle)\).

Hence, with probability \(p\), qubit \(A\) is in state \(\frac{1}{\sqrt{2}}(|0\rangle + |1\rangle)\), and with probability \(1 - p\) the qubit is in state \(\frac{1}{\sqrt{2}}(|0\rangle - |1\rangle)\), exactly as stated in Eq. (6.12).
Chapter 7

Einstein-Podolsky-Rosen (EPR), Bell’s inequalities, and Local Realism

7.1 Introduction

In classical physics, objects have definite properties irrespective of whether we measure them or not. This is called objective reality. A measurement just reveals a property which already existed.

However, this is not the case in quantum mechanics. To see this, suppose that a qubit is initially in the state

\[ |\psi\rangle = \frac{1}{\sqrt{2}} (|0\rangle + |1\rangle). \] (7.1)

If we measure the qubit (i.e. measure \( Z \)) the Born rule states that we get \(|0\rangle\) (i.e. eigenvalue +1) with probability \( \frac{1}{2} \) and \(|1\rangle\) (i.e. eigenvalue −1) with probability \( \frac{1}{2} \). However, we can not infer from this that, before the measurement, the qubit was in state \(|0\rangle\) with probability \( \frac{1}{2} \) and \(|1\rangle\) with probability \( \frac{1}{2} \), for this leads to a contradiction as we will now see.

If we apply the Hadamard operator,

\[ H = \frac{1}{\sqrt{2}} \begin{pmatrix} 1 & 1 \\ 1 & -1 \end{pmatrix} \] (7.2)

to \(|\psi\rangle\) we get

\[ H|\psi\rangle = |0\rangle. \] (7.3)

Hence, according to the Born rule if we measure a qubit in state \( H|\psi\rangle \), i.e. after applying the Hadamard, we get \(|0\rangle\) with probability 1.

However, suppose we assume that, before the measurement, the qubit in state \(|\psi\rangle\) corresponds to being in state \(|0\rangle\) with probability \( \frac{1}{2} \) and \(|1\rangle\) with probability \( \frac{1}{2} \), then the action of \( H \) on \(|\psi\rangle\) produces either \( \frac{1}{\sqrt{2}}(|0\rangle + |1\rangle) \) or \( \frac{1}{\sqrt{2}}(|0\rangle - |1\rangle) \), again with equal probability, and so, in either case, measurement of the qubit gives \(|0\rangle\) or \(|1\rangle\), again with equal probability. This is in contradiction to Eq. (7.3), which states that the measurement gives \(|0\rangle\) with probability 1. Hence we can not assume that state \(|\psi\rangle\) in Eq. (7.1) corresponds to being in \(|0\rangle\) with probability \( \frac{1}{2} \) and \(|1\rangle\) with probability \( \frac{1}{2} \).

One person who did not like that quantum mechanics describes a world without objective reality (and that quantum mechanics involves probabilities at a fundamental level), was Albert Einstein. In 1935 he wrote a famous paper with Podolsky and Rosen (now called EPR), in which they simply asserted that nature has the property of objective reality. According to this picture of the world, the

---

1He reputedly claimed to Niels Bohr that "God does not play dice with the universe". Bohr’s reported reply, which may be apocryphal, was “Albert, you shouldn’t tell God what to do".
reason that, in general, measurements do not give a definite answer but gives different results with various probabilities, is that quantum mechanics, as we have it, is incomplete. Rather, there is a deeper level of structure, which we don’t have access to at present, with extra, hidden, variables, such that if we could access those variables, the measurement would be deterministic and would just reveal the state of the system which existed previously, i.e. we would have objective reality. The fact that measurements on a quantum state do not give a unique result is, in this picture, because the hidden variables have different values when the different measurements are done.

The classical, EPR picture is called local realism:

1. **Realism.** The measured values of each particle are objectively real. They have definite values before measurement and irrespective of whether or not a measurement is made.

2. **Locality.** A measurement of \( A \) does not affect \( B \) instantaneously. More precisely, the measurement of \( A \) has no effect on \( B \) if \( A \) and \( B \) are spatially separated, i.e. \( |\vec{r}_A - \vec{r}_B| > ct \) where \( t \) is the time between measurements and \( c \) is the speed of light. This is just special relativity, one of Einstein’s greatest insights.

### 7.2 An EPR Experiment

In this chapter we will describe an experiment in which quantum mechanics gives different results from *any* local realistic theory. Such experiments have been done and found to be in agreement with quantum mechanics and in disagreement with local realism.

EPR examined a thought experiment with entangled particles. We shall consider a simpler version of the EPR thought experiment to Bohm. For this experiment we will derive a condition (an inequality) which any theory with local realism must have, but which is violated by quantum mechanics. This is one of many inequalities of a similar nature, the first of which was discovered by John Bell. Hence they are known quite generally as Bell’s inequalities.

We suppose that an experimenter prepares pairs of 2-state particles (qubits) in the following entangled Bell state

\[
|\psi\rangle = \frac{1}{\sqrt{2}} (|01\rangle - |10\rangle). \quad (7.4)
\]

He sends one particle of the pair to Alice and the other, in the opposite direction, to Bob, see Fig. 7.1. He then repeats this for many pairs. Suppose that Alice and Bob measure the particles in the computational \((Z)\) basis. If Alice measures \(|0\rangle\) (for which the eigenvalue of \(Z\) is +1) then Bob must measure the opposite, i.e. \(|1\rangle\) (for which the eigenvalue of \(Z\) is −1).

![Figure 7.1: Sketch of the experimental setup for the version of the EPR experiment discussed in the text. The source emits pairs of qubits in the state \(|\psi\rangle = \frac{1}{\sqrt{2}} (|01\rangle - |10\rangle)\) given in Eq. (7.4). For each pair Alice and Bob decide independently and randomly which of the three non-orthogonal directions, \(\vec{a}, \vec{b}\) or \(\vec{c}\), to measure along. The result in each case is +1 or −1. The double lines indicate that the result of the measurement is a classical bit.](image)

Now consider a general basis. As discussed in Chapter 4, a general qubit state \(|0_\hat{n}\rangle\) is characterized by two parameters, \(\theta\) and \(\phi\), which are the polar and azimuthal angles of a point in direction \(\hat{n}\) on the
unit sphere, known, in this context as the Bloch sphere, see Fig. 4.1. The state on the antipodal point on the sphere is denoted by $|\hat{n}\rangle$. The connection between $|0\rangle$ and $|1\rangle$ and the basis states in the computational basis, $|0\rangle$ and $|1\rangle$, is given by Eqs. (4.11).

It is shown in Eq. (7.29) in Appendix 7.A that Eq. (7.4) can equivalently be written as

$$|\psi\rangle = \frac{1}{\sqrt{2}} (|0\hat{n}\rangle - |1\hat{n}\rangle),$$

(7.5)

ignoring an overall phase, for any direction $\hat{n}$. Hence the state in Eq. (7.4) has the interesting property that Alice and Bob will always get opposite results as long as they measure in the same basis no matter what that basis is.

The results of the measurements of Alice and Bob are therefore strongly correlated. Of course, one can also have correlations between experimental results in classical systems. However, we will show below that the quantum correlations in entangled states like that in Eq. (7.5) are different from classical correlations.

In the experiment that we will consider, Alice and Bob can each choose to measure in one of three distinct, non-orthogonal directions $\vec{a}, \vec{b}$ and $\vec{c}$. Every time they receive a particle they separately choose at random one of these three directions and record whether they get +1 or −1.

The timing of the measurements is important. They must be done a causally disconnected manner so information about the direction that Alice has chosen cannot have reached Bob when he makes his measurement, and vice versa.

The setup is sketched in Fig. 7.1.

### 7.3 Bells’ Inequality

If Alice and Bob choose the same direction we know that they will get opposite results. Next consider in some detail what happens when Alice and Bob do not choose the same direction.

Firstly let us see what happens in a classical picture with objective reality.

The qubits then have a well defined state prior to the measurement. The reason that we don’t always get the same result for measurements along a given direction must be that the qubit pairs are not all emitted in the same state each time. Rather, each possible result of the measurements corresponds to a particular type of initial state. There are three directions, for each of which Alice and Bob get one of two possible results. Hence there are $3^2 = 8$ different sets of possible results, and hence 8 populations of initial states that we need to consider, as shown in Table 7.1. For the $i$-th type, $N_i$ pairs will be generated where

$$N = \sum_{i=1}^{8} N_i,$$

(7.6)

is the total number of pairs. To agree with experiment, we must assume that if Alice and Bob measure in the same direction they will get opposite results. This information is incorporated into Table 7.1.

---

2Note: when we say “measure the qubit in the $\hat{n}$ basis” we mean measure $\hat{\sigma} \cdot \hat{n}$, where, as discussed in Sec. 2.3, the $\sigma_\alpha, (\alpha = x, y, z)$ are just another notation for the Pauli operators $X, Y$ and $Z$.

3In the simpler setup of two directions, one finds that there is no incompatibility between quantum mechanics and local realism. Three directions is the minimum needed to derive an inequality which is violated by quantum mechanics.

4Since Alice and Bob always get the opposite result if they measure in the same direction there are only $2^3$, not $2^6$, different populations.

5The state $|\psi\rangle$ in Eq. (7.4) is known as a “spin singlet” state in the physics literature and has zero total spin angular momentum. Assuming that the initial state of the source, before the qubits are emitted, has zero angular momentum, then conservation of angular momentum requires that the qubits be emitted in state $|\psi\rangle$ and therefore that Alice and Bob must get opposite results if they measure in the same direction.
CHAPTER 7. EINSTEIN-PODOLSKY-ROSEN (EPR), BELL’S INEQUALITIES, AND LOCAL REALISM

<table>
<thead>
<tr>
<th>Population</th>
<th>Alice</th>
<th>Bob</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>a b c</td>
<td>a b c</td>
</tr>
<tr>
<td>$N_1$</td>
<td>+ + +</td>
<td>- - -</td>
</tr>
<tr>
<td>$N_2$</td>
<td>+ + -</td>
<td>- - +</td>
</tr>
<tr>
<td>$N_3$</td>
<td>+ - +</td>
<td>- + -</td>
</tr>
<tr>
<td>$N_4$</td>
<td>+ - -</td>
<td>- + +</td>
</tr>
<tr>
<td>$N_5$</td>
<td>- + +</td>
<td>+ - -</td>
</tr>
<tr>
<td>$N_6$</td>
<td>- + -</td>
<td>+ - +</td>
</tr>
<tr>
<td>$N_7$</td>
<td>- - +</td>
<td>+ + -</td>
</tr>
<tr>
<td>$N_8$</td>
<td>- - -</td>
<td>+ + +</td>
</tr>
</tbody>
</table>

Table 7.1: The eight populations of qubit pairs which give different results when measured along the $\vec{a}, \vec{b}$ and $\vec{c}$ directions. Note that Alice and Bob get opposite results if they measure in the same direction, so Bob’s side of the table is precisely the opposite of Alice’s. Hence there are only $2^3$ possible sets of outcomes.

For example, referring to Table 7.1 for a qubit pair in population 4, Alice will get +1 if she measures in direction $\vec{a}$, and Bob will get +1 if he measures in direction $\vec{b}$. Similarly for population 7, Alice will get −1 if she measures in direction $\vec{a}$ and Bob will get +1 if he measures in direction $\vec{b}$. In all cases, if Alice and Bob measure in the same direction they get opposite results.

We now make some simple observations. (Each observation is simple but one needs to focus to follow the thread of the argument to the end.) Clearly $N_i \geq 0$, so it must be true that

$$\frac{N_3 + N_4}{N} \leq \frac{N_2 + N_4}{N} + \frac{N_3 + N_7}{N}, \quad (7.7)$$

since $N_2$ and $N_7$, which can not be negative, have been added on the RHS.

- ($N_3, N_4$) According to Table 7.1 only for populations 3 and 4 would Alice get +1 if she measures along $\vec{a}$ and Bob would get +1 if he measures along $\vec{b}$. None of the other populations give this. Hence, among the times that Alice measures along $\vec{a}$ and Bob along $\vec{b}$, the probability that they both get +1 is $(N_3 + N_4)/N$. Let’s call this $P(+\vec{a}; +\vec{b})$, in which the first argument refers to Alice and the second to Bob, i.e.

$$\frac{N_3 + N_4}{N} = P(+\vec{a}; +\vec{b}). \quad (7.8)$$

- ($N_2, N_4$). Only for populations 2 and 4 would Alice get +1 measuring along $\vec{a}$ and Bob get +1 measuring along $\vec{c}$. Hence

$$\frac{N_2 + N_4}{N} = P(+\vec{a}; +\vec{c}). \quad (7.9)$$

- ($N_3, N_7$). Only for populations 3 and 7 would Alice get +1 measuring along $\vec{c}$ and Bob get +1 measuring along $\vec{b}$. Hence

$$\frac{N_3 + N_7}{N} = P(+\vec{c}; +\vec{b}). \quad (7.10)$$

Combining Eqs. (7.7)–(7.10), we have\(^6\)

$$P(+\vec{a}; +\vec{b}) \leq P(+\vec{a}; +\vec{c}) + P(+\vec{c}; +\vec{b}). \quad (7.11)$$

\(^6\)Recall what we mean by these probabilities. $P(+\vec{a}; +\vec{b})$, for example, means that, out of the times when Alice measured along $\vec{a}$ and Bob measured along $\vec{b}$, this is the probability that they both got +1.
If all the populations are equal, then each probability is 1/4 so the inequality is clearly satisfied. Equation (7.11) is an example of a Bell’s inequality. It is satisfied by any theory with local realism. Note that there is nothing sophisticated about this Bell’s inequality; it is just bookkeeping. I emphasize that Eq. (7.11) has nothing to do with quantum mechanics. In fact, we will now see that it is violated by quantum mechanics for a broad range of measurement directions $\vec{a}, \vec{b}, \vec{c}$.

We therefore now consider what quantum mechanics has to say.

The 2-qubit state generated by the source is given by Eq. (7.5) for any direction $\hat{n}$, where $|0\rangle_{\hat{n}}$ and $|1\rangle_{\hat{n}}$ are given by Eqs. (7.23). We take the $\theta = \phi = 0$ direction to be that of $\vec{a}$, so we write

$$|\psi\rangle = \frac{1}{\sqrt{2}} (|0\rangle_{\vec{a}}|0\rangle_{\vec{c}} - |1\rangle_{\vec{a}}|1\rangle_{\vec{c}}), \quad (7.12)$$

where we indicate on the RHS which qubit is meant (1 for Alice’s and 2 for Bob’s). We need the probability amplitude for this to be in the state with eigenvalue +1 along $\vec{a}$ for Alice and eigenvalue +1 along $\vec{c}$ for Bob, i.e. $|0\rangle_{\vec{a}}|0\rangle_{\vec{c}}$. Hence, to get $P(+\vec{a}; +\vec{c})$ we compute first the amplitude

$$<00|\psi\rangle = \frac{1}{\sqrt{2}} |0\rangle_{\vec{a}}|0\rangle_{\vec{c}}. \quad (7.14)$$

If $\vec{c}$ is at angles $(\theta_{ac}, \phi_{ac})$ relative to $\vec{a}$, then, according to Eq. (7.23a),

$$\frac{1}{\sqrt{2}} <0\rangle_{\vec{a}}|1\rangle_{\vec{a}} = \frac{1}{\sqrt{2}} e^{-i\phi_{ac}} \sin \frac{\theta_{ac}}{2}. \quad (7.15)$$

so

$$P(+\vec{a}; +\vec{c}) = |<00|\psi\rangle|^2 = \frac{1}{2} \left| e^{i\phi_{ac}} \sin \frac{\theta_{ac}}{2} \right|^2 = \frac{1}{2} \sin^2 \left(\frac{\theta_{ac}}{2}\right). \quad (7.16)$$

Equation (7.16) predicts that $P(+\vec{a}; +\vec{c}) \to 0$ if $\vec{a}$ and $\vec{c}$ are in the same direction. This result is correct because when Alice and Bob measure in the same direction they must get different results (in particular they can’t both get +1). Similarly

$$P(+\vec{a}; +\vec{b}) = \frac{1}{2} \sin^2 \left(\frac{\theta_{ab}}{2}\right), \quad (7.17)$$

$$P(+\vec{c}; +\vec{b}) = \frac{1}{2} \sin^2 \left(\frac{\theta_{cb}}{2}\right). \quad (7.18)$$

Hence Bell’s inequality, Eq. (7.11), when applied to quantum mechanics gives

$$\sin^2 \left(\frac{\theta_{ab}}{2}\right) \leq \sin^2 \left(\frac{\theta_{ac}}{2}\right) + \sin^2 \left(\frac{\theta_{cb}}{2}\right). \quad (7.19)$$

As we shall now see, it is easy to find cases where this is violated. Consider the situation in Fig. 7.2 and take $\theta = \pi/3$. We have

$$\sin^2 \left(\frac{\theta_{ac}}{2}\right) = \sin^2 \left(\frac{\theta_{cb}}{2}\right) = \sin^2 \left(\frac{\pi}{6}\right) = \frac{1}{4}, \quad (7.20)$$

and

$$\sin^2 \left(\frac{\theta_{ab}}{2}\right) = \sin^2 \left(\frac{\pi}{3}\right) = \frac{3}{4}. \quad (7.21)$$
Hence the LHS of Eq. (7.19) is $3/4$ while the RHS is $1/2$ so the inequality is violated. For general $\theta$ in Fig. 7.2, the inequality in Eq. (7.19) can be written

$$\sin \theta \leq \sqrt{2} \sin \left( \frac{\theta}{2} \right),$$

(7.22)

which is violated for $0 < \theta < \pi/2$, as shown graphically in Fig. 7.3.

Hence quantum mechanics violates Bell’s inequalities. These inequalities are satisfied by any theory with local realism. Experiments along the lines of that sketched in Fig. 7.1 have been done, using polarized photons. These experiments agree with quantum mechanics and disagree with local realism. See https://physics.aps.org/articles/v8/123 for a brief discussion of these experiments. Among the different experiments there are variations in the initial state of the entangled qubits and in which Bell’s inequality is being tested, but they are all equivalent. The more sophisticated experiments choose (randomly) the polarization directions while the photons are in flight. This makes it impossible for a measurement at one polarizer to depend on the chosen orientation of the other, since relativistic causality (i.e. the fact that information can not travel faster than the speed of light) prevents one polarizer from “knowing” the orientation of the other at the time of a measurement. This feature of the experiment is necessary to show that no local hidden variable theory can explain the data.

Bell’s inequalities characterize quantum correlations between two entangled qubits, which are different from classical correlations. Very recently non-classical correlations, distinct from those of Bell,
have been found in experiments with three sources of pairs of entangled photons and three detectors in
the shape of a triangle, see [link to experiment]. Thus the study of non-classical correlations in quantum mechanics, stimulated by EPR in the 1930s, made precise by Bell in the 1960s, and studied experimentally since the 1970s, remains an active field up to the present day.

Although the experimentally found violations of Bell’s inequalities rule out local theories with
objective reality, they do not, in principle, rule out non-local theories with objective reality. However,
these would violate special relativity which is one of the cornerstones of physics. Hence very few
physicists think that a non-local theory of quantum mechanics will turn out to be the correct theory
of nature.

Thus quantum mechanics is strange:

- Unlike in classical physics, probabilities enter in a fundamental way.
- Unlike in classical physics, we do not have objective reality. Reality is an emergent concept at
bigger scales. (Surely we all agree that the moon is there even when we don’t look at it.)

Many physicists feel uncomfortable with these aspects of quantum mechanics, and hope that a
better insight will emerge. But, in the 85 years since the EPR paper this has not happened, so we will
probably have to continue living with the strange world of quantum mechanics as we now understand
it.

Can we use the differences between the strange quantum world and the familiar classical world to
do more efficient computation, at least for some problems? This question will be the focus of the rest
of the course.

Appendix

7.A The spin-singlet state is isotropic

We showed in Eqs. (4.11) of Chapter 4 that the eigenstate of spin in a direction specified by polar
angles \((\theta, \phi)\) with eigenvalue \(+1\) is given by

\[
\begin{align*}
|0\rangle &= \cos \frac{\theta}{2} |0\rangle + e^{i\phi} \sin \frac{\theta}{2} |1\rangle, \\
|1\rangle &= -e^{-i\phi} \sin \frac{\theta}{2} |0\rangle + e^{i\phi} \cos \frac{\theta}{2} |1\rangle,
\end{align*}
\]

see Fig. 4.1 A similar calculation gives the eigenstate corresponding to eigenvalue \(-1\) to be

\[
|1\rangle = -\sin \frac{\theta}{2} |0\rangle + e^{i\phi} \cos \frac{\theta}{2} |1\rangle,
\]

which is the antipodal point where \(\theta \to \pi - \theta, \phi \to \phi + \pi\).

From Eqs. (7.23a) and (7.23b), we see that the unitary matrix which transforms from the \(Z\) basis
to the \(\hat{n}\) basis is

\[
U = \begin{pmatrix}
\cos \frac{\theta}{2} & e^{i\phi} \sin \frac{\theta}{2} \\
-\sin \frac{\theta}{2} & e^{i\phi} \cos \frac{\theta}{2}
\end{pmatrix}.
\]

The inverse transformation is given by \(U^{-1}\), but since \(U\) is unitary we have

\[
U^{-1} = U^\dagger \equiv (U^T)^* = \begin{pmatrix}
\cos \frac{\theta}{2} & -\sin \frac{\theta}{2} \\
e^{-i\phi} \sin \frac{\theta}{2} & e^{-i\phi} \cos \frac{\theta}{2}
\end{pmatrix},
\]

so

\[
\begin{align*}
|0\rangle &= \cos \frac{\theta}{2} |0\rangle - \sin \frac{\theta}{2} |1\rangle, \\
|1\rangle &= e^{-i\phi} \sin \frac{\theta}{2} |0\rangle + e^{-i\phi} \cos \frac{\theta}{2} |1\rangle.
\end{align*}
\]

The term non-local refers to information propagating faster than the speed of light.
We can now see what the entangled Bell state\footnote{In the physics literature this is called the spin-singlet state.} $|\psi\rangle$ in Eq. (7.4) looks in the $\hat{n}$ basis.

\begin{align}
|\psi\rangle &= \frac{1}{\sqrt{2}} (|01\rangle - |10\rangle) \\
&= \frac{1}{\sqrt{2}} \left[ (\cos \frac{\theta}{2} |0_{\hat{n}}\rangle_1 - \sin \frac{\theta}{2} |1_{\hat{n}}\rangle_1) \left( e^{-i\phi} \sin \frac{\theta}{2} |0_{\hat{n}}\rangle_2 + e^{-i\phi} \cos \frac{\theta}{2} |1_{\hat{n}}\rangle_2 \right) - \\
&\quad \left( e^{-i\phi} \sin \frac{\theta}{2} |0_{\hat{n}}\rangle_1 + e^{-i\phi} \cos \frac{\theta}{2} |1_{\hat{n}}\rangle_1 \right) \left( \cos \frac{\theta}{2} |0_{\hat{n}}\rangle_2 - \sin \frac{\theta}{2} |1_{\hat{n}}\rangle_2 \right) \right] \\
&= e^{-i\phi} \frac{1}{\sqrt{2}} (|0_{\hat{n}}1_{\hat{n}}\rangle - |1_{\hat{n}}0_{\hat{n}}\rangle),
\end{align}

where, in the middle expression, we indicated by a subscript, e.g. $|\cdots\rangle_1$, whether the state is that of the first or second qubit. Apart from the unimportant overall phase factor of $e^{-i\phi}$, Eq. (7.29) is of the same form as the state takes in the computational ($Z$) basis, Eq. (7.27). Hence if the two qubits are observed in the same basis (see footnote\textsuperscript{2} on page 55), no matter which one, the results of the two measurements will always be opposite, one giving $+1$ and the other $-1$.

\footnote{Note that $e^{-i\phi}$ is just the determinant of the transformation matrix from the computational basis to the $\hat{n}$ basis given in Eq. (7.25). Quite generally, if the “singlet” state $|\psi\rangle$ in Eq. (7.27) is acted on by a unitary transformation $V$ then one can show that $V|\psi\rangle = \det V |\psi\rangle$. Since $V$ is unitary its determinant can only be a pure phase.}
Chapter 8

Classical and Quantum Gates

Now, finally, we get to computation!

The elementary circuit elements which acts on the data in a computer are called gates. In this chapter we will first discuss classical gates and then go on to describe quantum gates.

8.1 Classical Gates

Data in a classical digital computer is in the form of bits, $x$, which take values 0 or 1. The only operation involving a single classical bit, i.e. the only 1-bit classical gate, is NOT which takes 0 to 1 and vice versa.

Of particular interest are 2 bit gates, the most common ones being

\[
\begin{array}{c|c|c}
\text{In} & \text{Out} \\
00 & 0 \\
01 & 0 \\
10 & 0 \\
11 & 1 \\
\end{array}
\]

\[
\begin{array}{c|c|c}
\text{In} & \text{Out} \\
00 & 0 \\
01 & 1 \\
10 & 1 \\
11 & 1 \\
\end{array}
\]

\[
\begin{array}{c|c|c}
\text{In} & \text{Out} \\
00 & 0 \\
01 & 1 \\
10 & 1 \\
11 & 0 \\
\end{array}
\]

These have two input bits and one output bit. For the AND gate the result is 0 unless both inputs are 1. For the OR gate the result is 0 unless one or both of the inputs are 1. The XOR gate only differs from the OR gate in giving zero if both the inputs are 1.

Note that AND gives the same results as multiplication of the bits $xy$. The XOR operation is equivalent to addition of the bits modulo 2, i.e. $x + y \pmod{2}$. To see this, note that the modulo
operation gives the remainder after integer division. For example, since $13 = (5 \times 2) + 3$ we have $13 \mod 5 = 3$. Referring to the XOR gate consider the case $x = y = 1$, so we have $1 + 1 \mod 2 = 0$, which is the value of XOR in this case. It is trivial to see that XOR is also addition modulo 2 for the other values of $x$ and $y$. For convenience of notation $x + y \mod 2$ is written as $x \oplus y$.

One can show that the AND, NOT and OR gates form a universal set which means that any logical operation on an arbitrary number of qubits can be expressed in terms of these gates. Thus, classically, we only need 1-bit and 2-bit gates to perform any operation.

However, we cannot directly take over gates like AND, OR and XOR to a quantum computer for the following reason. A gate in a quantum computer will be implemented by a unitary operator acting on a small number of qubits. A unitary operator has the property that $U^{-1} = U^\dagger$. Now $U^{-1}$ performs the inverse operation, and since $U^\dagger$ is well defined the inverse operation must exist. Thus, quantum gates must be reversible.

However, AND, OR and XOR can not be reversible because they have a different number of outputs and inputs. Suppose, for example, we know that the output from an OR gate is 1, and want to know what is the input. We can’t say because there are three possible inputs, 01, 10 and 11, which give this output.

Thus, a major change in going from classical to quantum computing will be having to deal with reversible computation. We will consider first reversible classical computation before doing the quantum case.

Clearly a necessary condition for a gate to be reversible is that it has the same number of input and output bits. The 1-bit NOT gate has one input and one output, and is reversible since acting twice gives back the original bit, i.e. $(\text{NOT})^2 = \text{IDENTITY}$, so $(\text{NOT})^{-1} = \text{NOT}$, i.e. NOT is its own inverse.

We will now consider a reversible, classical, 2-bit gate, the quantum analog of which will play an important role in quantum computing. This is the controlled-NOT, or CNOT gate. It is similar to XOR except that it has a second output bit, one of the input qubits which remains unchanged on output. As we shall see, this simple change suffices to make the gate reversible.

One way of representing the action of CNOT is

$$\begin{pmatrix} x \\ y \end{pmatrix} \rightarrow \begin{pmatrix} x \\ x \oplus y \end{pmatrix}.$$  \hfill (8.2)

The first (upper) bit is called the control bit. This is unchanged by the action of CNOT. The second (lower) bit is called the target bit, and the effect of the XOR operation $x \oplus y$ is to flip $y$ if $x = 1$ and to leave $y$ alone if $x = 0$. Hence, as far as the target bit is concerned, the gate is indeed a controlled NOT, since the NOT acts if $x$, the control bit, is 1, and does not act if $x = 0$. The truth table is as follows:

<table>
<thead>
<tr>
<th>$x$</th>
<th>$y$</th>
<th>$x'$</th>
<th>$y'$</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>0</td>
<td>1</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>1</td>
<td>1</td>
</tr>
</tbody>
</table>
| 1   | 1   | 1    | 0    | \hfill (8.3)

It is useful to represent the CNOT gate by a diagram, as shown in Fig. 8.1. The input is on the left and the output on the right. The upper line is the control bit, and has value $x$ on input, while the lower line is the target bit and has value $y$ on input. On output, the control qubit is unchanged and the target qubit is the exclusive or (XOR) of $x$ and $y$.

It is easy to see that CNOT is reversible since, if we act twice, we get back the original input because

$$\begin{pmatrix} x \\ y \end{pmatrix} \xrightarrow{\text{CNOT}} \begin{pmatrix} x \\ x \oplus y \end{pmatrix} \xrightarrow{\text{CNOT}} \begin{pmatrix} x \\ x \oplus x \oplus y \end{pmatrix} = \begin{pmatrix} x \\ y \end{pmatrix}.$$  \hfill (8.4)
8.2 QUANTUM GATES

The last line follows because $x \oplus x = 0$ since $0 + 0 = 0$ and $1 + 1 = 0 \pmod{2}$. Thus CNOT is its own inverse. It can therefore be regarded as a reversible version of XOR.

Note that to be reversible it is not required that the inverse operator is the same as the original operator, only that the inverse operator exists. However, it turns out that most quantum gates we consider will be their own inverse.

We mentioned above that the 1-bit (NOT) gate and a set of irreversible 2-bit gates (AND and OR) together form universal set for a classical computer, which means that any logical operation on an arbitrary number of bits can be constructed out of these gates. The question we now ask is whether 1-bit and 2-bit reversible gates are universal. The answer is no. Classically one also needs a 3-bit gate such as the Toffoli gate shown in Fig. 8.2 or the Fredkin gate shown in Fig. 8.3.

Amazingly we shall see that 3-qubit gates are not needed quantum mechanically. In fact it is possible build the Toffoli gate, for example, out of 1-qubit and 2-qubit gates, and we will discuss how to do this in Sec. 12.B. We shall see that quantum mechanics allows for a big range of 1-qubit gates, whereas we have already noted that classically the only 1-bit gate is NOT. It is this wide range of possibilities for 1-qubit gates that allows us to construct a quantum mechanical Toffoli gate out of 1-qubit and 2-qubit gates, whereas no such construction is possible using classical gates.

8.2 Quantum Gates

Following David Deutsch we represent the action of quantum gates by a circuit. The circuit comprises a set of qubits in some initial state, acted on by gates and ending up in a final state. Each qubit is
represented by a line in the circuit diagram and time runs from left to right, see e.g. Fig. 8.4.

![Circuit Diagram](image)

Figure 8.4: A schematic circuit with three qubits and two gates. Time runs from left to right. The initial state of the qubits is $|i_1\rangle \otimes |i_2\rangle \otimes |i_3\rangle$ and the final state is $|f_1\rangle \otimes |f_2\rangle \otimes |f_3\rangle$.

Sometimes we will indicate a set of $n$ qubits (called a register) compactly by a single line with a slash through it as follows: $\overline{n}$.

Quantum circuits have the following properties:

- There are no loops, because qubits can’t go back in time.
- Lines can’t splay out (fan out) because of the no-cloning theorem.
- Similarly lines can’t merge.
- Gates and circuits are linear. We evaluate the effect of the circuit on an initial state which is a computational basis state. However, if the initial qubits are in a superposition of computational basis states, then the final qubits, after the circuit has acted, are easily computed since they are simply the corresponding linear superposition of outputs for each of the computational basis state inputs.

Circuits have several gates acting in succession on a qubit and it is important to understand the order in which they act. Unfortunately, this can be confusing. By convention, in diagrams time is from left to right, so in the diagram

| $i$ \[ A \[ B \[ f$  

A (the leftmost gate) acts first then B. However, when writing operator expressions, these work from right to left, so, the above diagram corresponds to

$$|f\rangle = BA|i\rangle,$$

in which $A$ is on the right. You simply have to get used to this reversal of order when going from circuit diagrams to operator expressions.

Now we describe some commonly used quantum gates, recalling that quantum gates are unitary operators and so must be reversible.

Firstly we consider 1-qubit gates.

- NOT (corresponds to the Pauli $X$ operator)

  $$X|0\rangle = |1\rangle, \quad X|1\rangle = |0\rangle,$$

  $$X = \begin{pmatrix} 0 & 1 \\ 1 & 0 \end{pmatrix}. \quad \text{(8.6)}$$
8.2. QUANTUM GATES

- Phase flip (corresponds to the Pauli $Z$ operator)

$$Z|0\rangle = |0\rangle, \quad Z|1\rangle = -|1\rangle, \quad Z = \begin{pmatrix} 1 & 0 \\ 0 & -1 \end{pmatrix}. \quad (8.7)$$

In the physics literature $X$ and $Z$ are called Pauli spin matrices. There is also a third Pauli spin matrix $Y$ where

$$Y = iXZ = \begin{pmatrix} 0 & -i \\ i & 0 \end{pmatrix}. \quad (8.8)$$

- Hadamard

The Hadamard gate $H$ will be very important.

$$H = \frac{1}{\sqrt{2}}(X + Z) = \frac{1}{\sqrt{2}} \begin{pmatrix} 1 & 1 \\ 1 & -1 \end{pmatrix}. \quad (8.9)$$

Note that $H^2 = 1$, and similarly $X^2 = Y^2 = Z^2 = 1$. Now a matrix which squares to the identity has eigenvalues $\pm 1$. To see this note that if $\vec{x}$ is an eigenvector of $A$ with eigenvalue $\lambda$ then

$$A^2\vec{x} = A(A\vec{x}) = A\lambda\vec{x} = \lambda A\vec{x} = \lambda^2\vec{x}. \quad (8.10)$$

But if $A^2 = 1$ then it follows that $\lambda^2 = 1$ and so $\lambda = \pm 1$.

We need to become familiar with the action of $H$ on computational basis states. This is:

$$H|0\rangle = |+\rangle \equiv \frac{1}{\sqrt{2}}(|0\rangle + |1\rangle) \quad (8.11)$$

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

Combining these two equations, the action of $H$ on a computational basis state $|x\rangle$ is seen to be

$$H|x\rangle = \frac{1}{\sqrt{2}}\left(|0\rangle + (-1)^x|1\rangle\right), \quad (8.12)$$

for both values of $x$, 0 and 1.

A crucial point is that these gates are linear, and so they act in the same way on a superposition. For example:

$$H[\alpha|0\rangle + \beta|1\rangle] = \frac{\alpha}{\sqrt{2}}(|0\rangle + |1\rangle) + \frac{\beta}{\sqrt{2}}(|0\rangle - |1\rangle) = \left(\frac{\alpha + \beta}{\sqrt{2}}\right)|0\rangle + \left(\frac{\alpha - \beta}{\sqrt{2}}\right)|1\rangle. \quad (8.13)$$

We also need to consider measurement gates, in which a classical measurement of a qubit takes place. By convention, measurements are made in the computational basis. The Pauli spin matrices are defined such that $Z$ is diagonal, so we can also call the computational basis the $Z$-basis.

The result of the measurement is a classical bit. In the circuit diagrams we indicate a classical bit by a double line, and so a measurement gate is indicated as follows:

```
A measurement gate
```
Figure 8.5: A quantum CNOT gate. If the initial state of the qubits (on the left) is a computational basis state, then the action of the quantum CNOT gate is the same as that of the classical CNOT shown in Fig. 8.1. The upper line represents the control qubit and the lower line the target qubit.

Next we consider 2-qubit gates, the most important of which by far is the CNOT. We already met the classical CNOT gate in Fig. 8.1. In the quantum case, if initially the qubits are in a computational basis state, then the action of the CNOT is the same as classically, i.e. as shown in Fig. 8.5.

The CNOT gate has the matrix representation

\[
U_{\text{CNOT}} = \begin{pmatrix}
0 & 0 & 0 & 0 \\
0 & 1 & 0 & 0 \\
0 & 0 & 0 & 1 \\
0 & 0 & 1 & 0
\end{pmatrix}
\] (8.14)

In this tensor product the control qubit is the one to the left. The target qubit (to the right) is flipped if the control qubit is 1 (so, relative to the identity matrix, columns 3 and 4 are interchanged). We can also write

\[
U_{\text{CNOT}} \text{ in terms of } 2 \times 2 \text{ blocks as follows}
\]

\[
U_{\text{CNOT}} = \begin{pmatrix}
1 & 0 \\
0 & X
\end{pmatrix}
\] (8.15)

The quantum aspect appears if we input (on the left) a linear combination of basis states. Suppose, for example, we set the target (lower) qubit to \(|0\rangle\). Then if the control qubit is initially \(|0\rangle\) the final state of the 2-qubit system is \(|00\rangle\), because the target qubit is not flipped (we take the control qubit to be the left one). If the control qubit is initially \(|1\rangle\) then the final state of the 2-qubit system is \(|11\rangle\) because the target qubit is flipped. Hence, by linearity, if the initial state of the control qubit is the superposition \(\alpha|0\rangle + \beta|1\rangle\), then the final state of the 2-qubit system is \(\alpha|00\rangle + \beta|11\rangle\), see Fig. 8.6.

\[
\alpha|0\rangle + \beta|1\rangle
\]

\[
|0\rangle
\]

\[
\begin{pmatrix}
\alpha|0\rangle + \beta|1\rangle \\
\alpha|0\rangle + \beta|1\rangle
\end{pmatrix}
\]

Figure 8.6: The action of the CNOT gate when the upper (control) qubit is initially in a superposition \(\alpha|0\rangle + \beta|1\rangle\), and the lower (target) qubit is initially \(|0\rangle\). By linearity, the final state is \(\alpha\) times the result of inputting \(|0\rangle\) in the control qubit plus \(\beta\) times the result of inputting \(|1\rangle\), i.e. \(\alpha|00\rangle + \beta|11\rangle\).

Note that if \(\alpha = 0\) (so \(\beta = 1\) since \(|\alpha|^2 + |\beta|^2 = 1\)) or \(\alpha = 1\) (\(\beta = 0\)), the final state is a clone of the initial state of the control qubit. However, for a general input state, the final state of the two qubits, \(\alpha|00\rangle + \beta|11\rangle\), is not a clone of the initial state of the control qubit which would be \((\alpha|0\rangle + \beta|1\rangle) \otimes (\alpha|0\rangle + \beta|1\rangle) = \alpha^2|00\rangle + \alpha\beta|01\rangle + \beta\alpha|10\rangle + \beta^2|11\rangle\). Hence there is no violation of the no-cloning theorem which states that a general quantum state can not be cloned.
In this course, we will specify the action of a gate by its action on an initial computational basis state. If we denote a qubit by a Latin letter, e.g. \( |x \rangle \), we mean that this is a computational basis state and \( x \) takes values 0 or 1. General quantum states, i.e. superpositions of computational basis states, will be indicated by Greek letters, e.g. \( |\psi \rangle \).

As already mentioned above, we do not need 3-qubit gates for quantum computing. More precisely, the statement is that one can generate an arbitrary unitary transformation (to a specified level of accuracy) on an arbitrary number of qubits, using only CNOT and single-qubit gates. I do not prove this result but refer interested students to a more advanced text [NC00]. It is fortunate that we don’t need 3-qubit gates given the difficulty of making quantum circuits.

It is useful to mention here that one has to be careful when dealing with superpositions, and one’s initial intuition as to the final result may be incorrect. As an example, consider the circuit in Fig. 8.7.

![Circuit Diagram](image)

Figure 8.7: The initial state of both qubits is \( |0 \rangle \). What is the final state \( |\psi_3 \rangle \)? Equation (8.16) gives the state of the two qubits at each stage. The end result is that the two qubits are entangled and, in contrast to what one might have thought, the control qubit has a non-zero amplitude to be flipped relative to its initial state, i.e. to be in state \( |1 \rangle \).

Since \( H^2 = 1 \) and the CNOT gate doesn’t change the control (upper) qubit, one might think that the final state of the control qubit would be the same as the initial state, i.e. \( |0 \rangle \). However this is not correct because the control and target qubits become entangled. Let’s go through each stage of the circuit using the notation in Fig. 8.7, and taking the left-hand qubit in the formulae to be the control qubit:

\[
\begin{align*}
|\psi_0 \rangle &= |00 \rangle \\
|\psi_1 \rangle &= \frac{1}{\sqrt{2}} (|00 \rangle + |10 \rangle ) \\
|\psi_2 \rangle &= \frac{1}{\sqrt{2}} (|00 \rangle + |11 \rangle ) \\
|\psi_3 \rangle &= \frac{1}{2} (|00 \rangle + |10 \rangle + |01 \rangle - |11 \rangle ) \\
&= \left[ |0 \rangle_c \otimes \left( \frac{|0 \rangle_t + |1 \rangle_t}{\sqrt{2}} \right) + |1 \rangle_c \otimes \left( \frac{|0 \rangle_t - |1 \rangle_t}{\sqrt{2}} \right) \right],
\end{align*}
\]

where in the last expression we indicate explicitly which qubit is the control qubit (“c”), and which the target qubit (“t”). We see that, contrary to what one might have initially guessed, there is an amplitude for the control qubit to be in state \( |1 \rangle \) because of its entanglement with the target qubit.

We have noted that the Pauli operators \( X, Y \) and \( Z \), and the Hadamard operator have eigenvalues \( \pm 1 \). Later in the course, when we consider the important topic of quantum error correction, we will encounter combinations of these operators on different qubits which also have \( \pm 1 \) eigenvalues. We will now describe a convenient way of measuring such operators. Let us denote the operator by \( U \). It will have an eigenvalue +1 with eigenvector \( |\psi_+ \rangle \) and an eigenvalue -1 with eigenvector \( |\psi_- \rangle \). We would
like to investigate the qubit (or qubits) to determine which eigenstate of $U$ it is in, or, if it is in a linear superposition, to project by measurement on to one of the eigenstates, and know which one.

Figure 8.8: A circuit with a control-$U$ gate in which the control (upper) qubit is surrounded by Hadamards. $U$ is an operator with eigenvalues $\pm 1$ and corresponding eigenvectors $|\psi_+\rangle$ and $|\psi_-\rangle$. As shown in the text, if a measurement of the upper qubit gives $|0\rangle$ then the lower qubit will be in state $|\psi_+\rangle$, and if the measurement gives $|1\rangle$ then the lower qubit will be in state $|\psi_-\rangle$. The states $|\phi_i\rangle (i = 0, 1, 2, 3)$ are described in the text.

A convenient way is to use the circuit shown in Fig. 8.8, which has a control-$U$ gate. If the control qubit is 1 the effect on the target qubit is

$$U|\psi_+\rangle = |\psi_+\rangle, \quad U|\psi_-\rangle = -|\psi_-\rangle.$$  \hfill (8.17)

If the control qubit is 0 then the target qubit is unchanged.

The lower (target) qubit is initially in state $|\psi\rangle$, which can be written as a linear combination of the two eigenvectors

$$|\psi\rangle = \alpha_+|\psi_+\rangle + \alpha_-|\psi_-\rangle,$$  \hfill (8.18)

and so, including the upper (control) qubit which is initially in state $|0\rangle$, the initial state of the circuit (on the left of Fig. 8.8) is

$$|\phi_0\rangle = \alpha_+|0 \psi_+\rangle + \alpha_-|0 \psi_-\rangle.$$  \hfill (8.19a)

After the first Hadamard on the upper qubit the state is

$$|\phi_1\rangle = \frac{\alpha_+}{\sqrt{2}} (|0 \psi_+\rangle + |1 \psi_+\rangle) + \frac{\alpha_-}{\sqrt{2}} (|0 \psi_-\rangle + |1 \psi_-\rangle).$$  \hfill (8.19b)

The effect of the control-$U$ gate on the target qubit is given by Eq. (8.17) when the control qubit is 1 and has no effect if the control qubit is 0. Hence, after the control-$U$ gate, the state is

$$|\phi_2\rangle = \frac{\alpha_+}{\sqrt{2}} (|0 \psi_+\rangle + |1 \psi_+\rangle) + \frac{\alpha_-}{\sqrt{2}} (|0 \psi_-\rangle - |1 \psi_-\rangle).$$  \hfill (8.19c)

Applying the second (rightmost) Hadamard to the upper qubit we get

$$|\phi_3\rangle = \alpha_+|0 \psi_+\rangle + \alpha_-|1 \psi_-\rangle.$$  \hfill (8.19d)

Hence if a measurement of the upper qubit gives $|0\rangle$ (which it does with probability $|\alpha_+|^2$) the lower qubit will be in state $|\psi_+\rangle$, and if the measurement gives $|1\rangle$ (probability is $|\alpha_-|^2$) the lower qubit will be in state $|\psi_-\rangle$. We see that measuring the control qubit projects the target qubit onto an eigenstate of $U$ and tells us which one.

We will return to the circuit in Fig. 8.8 later in the course when we discuss quantum error correction.\footnote{Apart from the absence of the final measurement gate, Fig. 8.8 is a special case of Fig. 8.8 with $U = X$ (since the NOT gate is effected by the $X$ operator).}
Chapter 9

Generating and measuring Bell States

Entangled states play an important role in quantum computing. The most-studied entangled states are so-called Bell states which involve two qubits. They are named in honor of the physicist John Bell who clarified the Einstein-Podolsky-Rosen (EPR) paradox, and whose inequalities demonstrated that the description of nature provided by quantum mechanics is fundamentally different from the classical description, see Chapter 7. The Bell states are defined by

\[
|\beta_{00}\rangle = \frac{1}{\sqrt{2}} \left( |00\rangle + |11\rangle \right),
\]

(9.1a)

\[
|\beta_{01}\rangle = \frac{1}{\sqrt{2}} \left( |01\rangle + |10\rangle \right),
\]

(9.1b)

\[
|\beta_{10}\rangle = \frac{1}{\sqrt{2}} \left( |00\rangle - |11\rangle \right),
\]

(9.1c)

\[
|\beta_{11}\rangle = \frac{1}{\sqrt{2}} \left( |01\rangle - |10\rangle \right).
\]

(9.1d)

These four equations can be combined as follows:

\[
|\beta_{xy}\rangle = \frac{1}{\sqrt{2}} \left( |0y\rangle + (-1)^x |1\bar{y}\rangle \right),
\]

(9.2)

where \( \bar{y} \) is the complement of \( y \), i.e. \( \bar{y} = 1 - y \).

![Circuit to create the Bell states defined by Eqs. (9.1)](image)

\[ \text{Figure 9.1: Circuit to create the Bell states defined by Eqs. (9.1). In the CNOT gate the upper qubit } |x\rangle \text{ is the control qubit and the lower qubit } |y\rangle \text{ is the target qubit.} \]

The Bell states are clearly entangled. They can be created out of two (unentangled) qubits in computational basis states \( |xy\rangle \) by the circuit shown in Fig. 9.1. To see this note that after the Hadamard the state is

\[
|xy\rangle \rightarrow \frac{1}{\sqrt{2}} \left( |0y\rangle + (-1)^x |1\bar{y}\rangle \right).
\]

(9.3)
The effect of the CNOT gate is to flip $y$ in the second term (since $x = 1$ there) and so we get Eq. (9.2)\footnote{The reason that $y$ in the Bell state, Eq. (9.2), changes to $y$ in the second term in Eq. (9.4) is because $x = 1$ and so the $y$ (target) qubit is flipped.}.

The circuit in Fig. 9.1 converts the computational basis to the Bell basis. The reverse of this circuit can be used to convert the Bell basis back to the computational basis as shown in Fig. 9.2. The measured values of $x$ and $y$ tell us which Bell state we started with. This is called a Bell Measurement.

To see that this works note that after the CNOT gate the state of the two qubits in Fig. 9.2 is

$$\frac{1}{\sqrt{2}} [ |0y\rangle + (-1)^x |1y\rangle ], \quad (9.4)$$

which is separable and so can be written as

$$\frac{1}{\sqrt{2}} [ |0\rangle + (-1)^x |1\rangle ] \otimes |y\rangle. \quad (9.5)$$

Recall that the left-hand qubit is the upper (control) qubit in Fig. 9.2 and the right hand qubit is the lower (target) qubit. Acting with the Hadamard has the effect

$$H \frac{1}{\sqrt{2}} [ |0\rangle + (-1)^x |1\rangle ] = |x\rangle, \quad (9.6)$$

so the final state in Fig. 9.2 is $|xy\rangle$ as desired.

Note that the Bell states $|\beta_{xy}\rangle$ provide a basis for two qubits, see Chapter 4 since they are normalized, mutually orthogonal and linearly independent. Consequently, if the state inputted into the Bell measurement circuit in Fig. 9.2 is not a single Bell state, but rather a linear combination,

$$|\psi_{in}\rangle = \sum_{x, y=0}^{1} \alpha_{xy} |\beta_{xy}\rangle, \quad (9.7)$$

with $\sum_{x, y} |\alpha_{xy}|^2 = 1$, then the probability that the measurements obtain a particular set of values for $x$ and $y$ is $|\alpha_{xy}|^2$. 

Figure 9.2: Circuit for Bell measurements. This will be used later in the course when we discuss teleportation.
Chapter 10

Quantum Functions

10.1 An elementary quantum function

In computation we need to evaluate functions. How can we do this in a quantum computer where functions are determined by unitary transformations which are reversible?

Let us first consider the simplest case, where the argument of the function, \( x \), is a single bit, and the result of the function, \( f(x) \), is also a single bit. In other words, \( x \) takes only the values 0 and 1, and the same for \( f(x) \). We need to have a qubit for \( x \) and an additional qubit which contains information on the function \( f(x) \).

The function \( f(x) \) will be implemented by a unitary operator \( U_f \) acting on two qubits such that

\[
U_f|x\rangle|y\rangle = |x\rangle|f(x) \oplus y\rangle.
\] (10.1)

Note the similarity with the CNOT gate, which is precisely of this form with \( f(x) = x \). It is easy to see that \( U_f^2 = 1 \) since

\[
U_f^2|x\rangle|y\rangle = U_f|x\rangle|f(x) \oplus y\rangle = |x\rangle|f(x) \oplus f(x) \oplus y\rangle = |x\rangle|y\rangle
\] (10.2)

since, as discussed earlier in the course, \( f(x) \oplus f(x) = 0 \). Hence \( U_f \) has an inverse, which is \( U_f \) itself.

The corresponding circuit diagram is shown in Fig. 10.1.

![Figure 10.1: Schematic diagram of a unitary transformation \( U_f \) for a function \( f(x) \) in which both the argument \( x \) and the function just take two values, 0 and 1.](image)

For a general function, the range of inputs can be represented by \( n \) bits, say, and the range of outputs by \( m \) bits. Thus we need a total of \( n + m \) qubits both in the initial state and final state. The unitary transformation is

\[
U_f|x\rangle_n|y\rangle_m = |x\rangle_n|f(x) \oplus y\rangle_m,
\] (10.3)

\(^1\)We need to have two qubits in both the initial and final states in order that the function is reversible, just as we needed two qubits in the final state as well as the initial state to make the CNOT gate, a reversible generalization of the XOR gate, see Chapter 8.
where the modulo 2 addition, indicated by $\oplus$, applies separately to each of the $m$ bits of $f(x)$ and $y$. The proof that $U_f$ is its own inverse is the same as that in Eq. (10.2). The circuit diagram corresponding to Eq. (10.3) is shown in Fig. 10.2.

Figure 10.2: Schematic diagram of a general unitary transformation $U_f$ for an $n$-bit input $x$ and an $m$-bit output $f(x)$. The upper register in the figure has $n$ qubits and contains the input value $x$. The lower register has $m$ qubits and contains information about the function value $f(x)$. The registers are shown as single lines. To ensure the transformation is reversible there are $n + m$ qubits in both the initial state (to the left) and final state (to the right).

One sometimes calls the upper register in Fig. 10.2 the “input” register, because it contains the input, $x$, and the lower register the “output register” because it contains information on the function $f(x)$. However, since both registers are present in the initial state (on the left) and the final state (on the right) this terminology can be confusing.

Note that if $y = 0$ the lower register contains precisely the function $f(x)$.

10.2 Quantum Parallelism

Things get interesting if we feed in a superposition. We can generate a uniform superposition by acting with Hadamards on $|0\rangle^n$. Note that for one qubit

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

and similarly applying a Hadamard to each of two qubits

$$H|0\rangle \otimes H|0\rangle = \frac{1}{2}(|0\rangle + |1\rangle) \otimes (|0\rangle + |1\rangle)$$

$$= \frac{1}{2}(|00\rangle + |01\rangle + |10\rangle + |11\rangle)$$

$$= \frac{1}{2}(|00\rangle_2 + |01\rangle_2 + |10\rangle_2 + |11\rangle_2) = \frac{1}{2} \sum_{x=0}^{3} |x\rangle_2.$$  \hspace{1cm} (10.5)

Here we have used the convenient notation that $|x\rangle_n$ is an $n$-qubit state in which the values of the individual qubits are the bits of the $n$-bit integer $x$. Generalizing we have

$$H^\otimes n |0\rangle_n = \frac{1}{2^{n/2}} \sum_{x=0}^{2^n-1} |x\rangle_n.$$  \hspace{1cm} (10.6)

Now lets consider the circuit shown in Fig. 10.3. The initial state is

$$|\phi_0\rangle = |0\rangle_n |0\rangle_m.$$  \hspace{1cm} (10.7)
so the state fed into the unitary operator $U_f$ is the superposition

$$|\phi_1\rangle = \frac{1}{2^{n/2}} \sum_{x=0}^{2^n-1} |x\rangle_n |0\rangle_m.$$  \hspace{1cm} (10.8)

Noting that the lower register is initialized to $|0\rangle$, then by linearity, the final state must be

$$|\phi_2\rangle = \frac{1}{2^{n/2}} \sum_{x=0}^{2^n-1} |x\rangle_n |f(x)\rangle_m.$$  \hspace{1cm} (10.9)

This is an astonishing result. The final state contains the function values for all $2^n$ possible values of the input $x$. They have been evaluated in parallel, a feature of quantum mechanics called, naturally enough, “quantum parallelism”. For $n = 100$ we have $2^{100} \approx 10^{30}$ function evaluations in parallel.

A speedup of $10^{30}$ seems to good to be true, and, unfortunately, it is. What’s the catch? The catch is that the only way one can access the information contained in the state is to do a measurement. This does not give $10^{30}$ results but just one result, the probabilities of the different results being the square of the amplitudes (which are all equal here). So, it seems that we have achieved nothing. We have found the value of the function for one value of its argument, which we could have got much more easily on a classical computer. However, for some problems, one can gain enough useful information to get a “quantum speedup” by doing clever pre-processing before the measurement. How to achieve this in practice will occupy us for most of the rest of the course.

Philosophers, and some physicists, debate whether one can really state that all $2^n$ values of the function have been evaluated since one can not observe them. Most physicists would argue that the only “real” quantities are those that can be observed, and, in particular, the quantum mechanical state itself is not real. From this point of view, it is not valid to claim that all $2^n$ values of the function have actually been evaluated.

Now we have done enough preliminaries to study our first quantum algorithm! This will be described in the next chapter.
Chapter 11

Deutsch’s Algorithm

11.1 Introduction

We now turn to our first algorithm, due to David Deutsch\(^1\) (1985) which is generally felt to have started\(^2\) the field of quantum computing.

As we shall see the problem is very trivial. It concerns functions which takes a 1-qubit argument and give a 1-qubit output. The problem is clearly contrived and is of no practical interest. However, it does show a quantum speedup, and this arises from the same features of quantum circuits, namely *quantum parallelism* and *interference*, used in more sophisticated and useful quantum algorithms such as that of Shor.

Since the input takes one of two values, 0 and 1, as does the output, there are only four distinct functions as shown in the table.

<table>
<thead>
<tr>
<th></th>
<th>(x = 0)</th>
<th>(x = 1)</th>
</tr>
</thead>
<tbody>
<tr>
<td>(f_1)</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>(f_2)</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>(f_3)</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>(f_4)</td>
<td>1</td>
<td>1</td>
</tr>
</tbody>
</table>

Table 11.1: The four functions which have a 1-qubit input and a 1-qubit output.

You see that \(f_1\) and \(f_4\) gave the same result for each input, they are *constant*. On the other hand, \(f_2\) and \(f_3\) give *different* results for the two inputs. This is analogous to a coin toss. The two values of \(x\) correspond to the two sides of the coin, the upper and the lower sides. The function values correspond to the result of the coin toss, heads or tails. If the two sides of the coin give different results (one heads and the other tails), corresponding to a non-constant function, the coin is honest. However, if the two sides of the coin give the same result (both heads or both tails) the coin is dishonest, corresponding to a constant function.

We are given a “black box”\(^3\) function \(f(x)\) and we want to learn about it. Of course we could just feed in \(x = 0\) and \(x = 1\) and observe the results. Suppose, however, we only want to know whether

---

\(^1\)We will actually present an improved version published somewhat later.

\(^2\)There was an earlier paper by Feynman (1982), but he was talking about using a quantum device to simulate a quantum system, i.e. a *simulator* rather than a computer. This field is very promising and we will likely have useful results from quantum simulators soon. However, quantum simulation is outside the scope of this course.

\(^3\)The term “black box” implies that the only information we can get about the function is by evaluating it for different inputs. We can’t open up the box to see what is inside.
the function is constant (satisfied by \( f_1 \) and \( f_4 \)) or not constant (satisfied by \( f_2 \) and \( f_3 \)). On a classical computer the only thing to do is to evaluate the function for both values of \( x \) and compare them, i.e. we need to make two calls to the function. However, we shall see that we can answer this question on a quantum computer with only one call to the function. We get less information than classically, because we don’t determine the individual values of \( f(0) \) and \( f(1) \), but we do determine whether or not \( f \) is constant. Hence Deutsch’s problem may be thought of as determining whether a coin to be tossed is honest or not with just one toss of the coin.

As we discussed in Chapter 10, a quantum function \( f \) is implemented by a unitary operator \( U_f \) as shown in Fig. 11.1. You will recall that we need to have the same number of qubits in the initial and finite states in order to make the function reversible. The function \( U_f \) in Fig. 11.1 is its own inverse as can be seen since \( U_f^2 = 1 \), the identity. If this is not obvious note that \( y \oplus f(x) \oplus f(x) = y \). (Adding 0 to 0 gives 0 and adding 1 to 1 mod 2 also gives 0.)

\[
\begin{array}{c}
|x\rangle \\
U_f \\
|y\rangle \\
\end{array} \quad \begin{array}{c}
|x\rangle \\
|y\oplus f(x)\rangle \\
\end{array}
\]

Figure 11.1: The blackbox routine \( U_f \) for a function \( f(x) \) which takes a 1-qubit input \( x \) and computes a 1-qubit function \( f(x) \). Here \( x \) and \( y \) are computational basis states \( |0\rangle \) or \( |1\rangle \). However, to gain a quantum speedup, we will input superpositions, generated by Hadamard gates, as shown in Fig. 11.2. We obtain the result of inputing a superposition from the results of inputing computational basis states by using linearity. Recall that time runs from left to right in circuit diagrams.

In order to take advantage of quantum parallelism we insert Hadamard gates before the black box function \( U_f \) on both the upper (input) and lower (output) qubits, and to take advantage of quantum interference of the results we will also put Hadamards on both qubits after \( U_f \) has acted, see Fig. 11.2. We initialize the upper qubit to be \( |0\rangle \) and the lower qubit to be \( |1\rangle \). The upper qubit could be initialized to either \( |0\rangle \) or \( |1\rangle \) but it is essential to initialize the lower qubit to \( |1\rangle \) as we shall see.

Recalling that
\[
H|0\rangle = \frac{1}{\sqrt{2}} (|0\rangle + |1\rangle), \quad H|1\rangle = \frac{1}{\sqrt{2}} (|0\rangle - |1\rangle),
\]
we find that after the first Hadamards the state in Fig. 11.2 is
\[
|\psi_0\rangle = \frac{1}{2} (|0\rangle + |1\rangle) \otimes (|0\rangle - |1\rangle),
\]
where, in the tensor product, the upper qubit is to the left and the lower qubit is to the right.

The function \( U_f \) is then applied. Recall from Fig. 11.1 that the final state of the lower qubit is \( f(x) \) if the initial of the lower qubit is zero, and the complement \( \overline{f(x)} \) if the initial state is one. Hence, after \( U_f \) has been applied, the state is
\[
|\psi_1\rangle = \frac{1}{2} |0\rangle \otimes (|f(0)\rangle - \overline{f(0)}) + \frac{1}{2} |1\rangle \otimes (|f(1)\rangle - \overline{f(1)})
\]
(11.3)

It is helpful to note that
\[
|f(x)\rangle - |\overline{f(x)}\rangle = \begin{cases} 
|0\rangle - |1\rangle & \text{if } f(x) = 0, \\
|1\rangle - |0\rangle & \text{if } f(x) = 1,
\end{cases}
\]
\[
= (-1)^{f(x)}(|0\rangle - |1\rangle).
\]
(11.4)
11.1. INTRODUCTION

Figure 11.2: Circuit for Deutsch’s algorithm. The initial state (on left) has $|0\rangle$ in the upper (input) qubit and $|1\rangle$ in the lower (output) qubit. Hadamard gates are applied to both qubits both before and after the function $U_f$ (which we assume to be an unknown black box). In the final state the lower qubit is unchanged at $|1\rangle$. A measurement is made of the final value (on right) of the upper qubit. If this is unchanged, i.e. $x = 0$ in this case, then the function is constant, while if the upper qubit has flipped, then the function is not constant. One could equivalently start with the upper qubit as $|1\rangle$ and find the same conclusion: if the upper qubit is unchanged the function is constant whereas if it has flipped the function is not constant.

Hence whether or not $f(x) = 0$ or $f(x) = 1$ just changes the overall sign of the state. To get this effect it was necessary to prepare the lower qubit in state $|1\rangle$ rather than $|0\rangle$. Vathsan [Vat16] calls Eq. (11.4) “phase kickback”. Consequently we can write $|\psi_1\rangle$ as

$$|\psi_1\rangle = \frac{(-1)^{f(0)}|0\rangle + (-1)^{f(1)}|1\rangle}{\sqrt{2}} \otimes \frac{|0\rangle - |1\rangle}{\sqrt{2}}. \quad (11.5)$$

Now we run both qubits through Hadamards (those to the right of $U$ in Fig. 11.2). It is easy to see that action on the lower qubit (right hand one in the tensor product) is to convert $\frac{1}{\sqrt{2}}(|0\rangle - |1\rangle)$ back to $|1\rangle$. The action of $H$ on the upper qubit (left hand one in the tensor product) is to give

$$\frac{1}{2} \left[ (-1)^{f(0)}(|0\rangle + |1\rangle) + (-1)^{f(1)}(|0\rangle - |1\rangle) \right] \quad (11.6)$$

which can be written as

$$\frac{1}{2} |0\rangle \left[ (-1)^{f(0)} + (-1)^{f(1)} \right] + \frac{1}{2} |1\rangle \left[ (-1)^{f(0)} - (-1)^{f(1)} \right]. \quad (11.7)$$

Clearly this is $\pm|0\rangle$ if $f(0) = f(1)$ (where the plus sign is for $f(0) = f(1) = 0$ and the minus sign for $f(0) = f(1) = 1$), and is $\pm|1\rangle$ if $f(0) \neq f(1)$ (where the sign depends on whether $f(0) = 1, f(1) = 0$ or vice versa). Hence the state to the right of the Hadamards in Fig. 11.2 is

$$|\psi_2\rangle = \begin{cases} 
\pm|0\rangle \otimes |1\rangle & \text{if } f(1) = f(0) \\
\pm|1\rangle \otimes |1\rangle & \text{if } f(1) \neq f(0)
\end{cases} \quad (11.8)$$

Consequently, measuring the upper qubit in Fig. 11.2 (left in the tensor product) tells us that $f(0) = f(1)$ if it is unchanged from the initial state, i.e. if we get $|0\rangle$, and $f(0) \neq f(1)$ if the upper qubit is flipped, i.e. if we get $|1\rangle$. We do this with one call to the function so we have achieved a “quantum speedup” of 2, which is admittedly not spectacular but it is interesting that we do get a speedup. We will do better later.
If we could measure the sign of the state we could determine the values of \( f(0) \) and \( f(1) \) separately but the sign of the state (more generally its phase) has no effect and can not be determined.

A crucial role has been played by the Hadamards. Those which act before \( U \) is called generate a superposition state with both inputs \( x = 0 \) and \( 1 \) present. Looking at Eq. (11.3) it “seems” that \( U \) has computed \( f(x) \) for both values of \( x \) with just one call to it. This is “quantum parallelism”. If we do a measurement directly after the application of \( U \) we only get one value. However, for certain problems like this one, if we do some additional post-processing (in this case acting with Hadamards again), we can use “quantum interference” between the different pieces in the superposition to get useful information (in this case whether the function is constant or not) when the measurement is subsequently done.

Note that the Deutsch algorithm is not probabilistic: it succeeds with probability 1. This shows that quantum algorithms don’t necessarily have to be probabilistic (though many are). In this case, quantum interference transforms the state to be measured into an eigenstate of the computational basis. As we know, if we measure an eigenstate we always get the same answer (the eigenvalue) and there is no uncertainty.

Appendices

11.A An alternative derivation

To familiarize ourselves with quantum circuits we will obtain Eq. (11.8) in a different way by explicitly writing down circuits for the four functions \( f_1 \) to \( f_4 \), see Fig. 2.1 of Mermin [Mer07]. Noting that the function flips the lower (output) qubit if the result of the function is 1 but leaves it alone if the function gives 0, we can represent the four functions in Table 11.1 by the circuits shown in Fig. 11.3.

We sandwich each of these functions between Hadamards to carry out the Deutsch algorithm, as shown in Fig. 11.2 and prepare the qubits in the initial state \( |x⟩ \otimes |1⟩ \). The results are shown in Fig. 11.4. It is important to understand each of the diagrams.

- **\( f_1 \):**
  This follows simply because \( U_{f_1} \) makes no change, see Fig. 11.3 and \( H^2 = 1 \) (the identity), see Fig. 11.5(a) in Appendix 11.B so the final qubits are the same as the initial qubits, \( |x⟩ \otimes |1⟩ \), see Fig. 11.4. In particular \( x \) is unchanged indicating, correctly that the function is constant.

- **\( f_2 \):**
  The function \( U_{f_2} \) has a CNOT gate in which the upper qubit is the control and the lower qubit is the target, see Fig. 11.3. The result of sandwiching a CNOT between Hadamards is, perhaps surprisingly, to interchange the role of the target and control qubits. This is shown in Appendix 11.B see Fig. 11.5(f). Hence we see that \( x \) is flipped because the lower qubit is set to \( |1⟩ \), see Fig. 11.4. This is correct because the function is not constant.

- **\( f_3 \):**
  The derivation of the equivalent circuit shown for \( U_{f_3} \) shown in Fig. 11.3 uses the result for \( U_{f_2} \) and, in addition, that \( HXH = Z \), see Fig. 11.5(b) in the Appendix 11.B Hence \( x \) is flipped and there is a sign change, see Fig. 11.4. We can’t measure the sign change but the fact that \( x \) is flipped correctly indicates that the function is not constant.

- **\( f_4 \):**
  The function \( U_{f_4} \) has an \( X \) gate on the lower qubit, see Fig. 11.3 and again we have \( HXH = Z \). Hence \( x \) remains unchanged and there is a sign change, see Fig. 11.4. Again we cannot measure the sign change and the fact that \( x \) is not flipped indicates correctly that the function is constant.
11.B Derivation of some useful identities in quantum circuits

We have

\[ X = \begin{pmatrix} 0 & 1 \\ 1 & 0 \end{pmatrix}, \quad Z = \begin{pmatrix} 1 & 0 \\ 0 & -1 \end{pmatrix}, \quad H = \frac{1}{\sqrt{2}} \begin{pmatrix} 1 & 1 \\ 1 & -1 \end{pmatrix}. \]  
(11.9)

By direct calculation it is easy to see that \( X^2 = 1, Z^2 = 1, \) and

\[ H^2 = 1, \]  
(11.10)

where 1 is the identity

\[ 1 = \begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix}. \]  
(11.11)

Equation (11.10) is represented graphically by Fig. 11.5(a). Also by direct calculation, we have \( XH = HZ. \) Hence multiplying on the left by \( H \) gives

\[ HXH = Z, \]  
(11.12)

see Fig. 11.5(b) for a graphical illustration, and multiplying on the right by \( H \) gives

\[ HZH = X, \]  
(11.13)

which is illustrated graphically in Fig. 11.5(c)
The NOT part of the CNOT gate is performed by the X operator. Hence it is sometimes convenient to denote a CNOT gate as a control-X gate, see Fig. 11.3(d). We will also meet the control-Z gate, in which the target qubit is acted upon by Z if the control qubit is 1, and otherwise the target qubit is unchanged. As with the control-X gate, there is no change in the control qubit. With a bit of thought, we see that the only effect of the control-Z gate is to change the overall sign of the state if both the target and control qubits are one. Thus the distinction between target and control is non-existent, and control and target qubits can be interchanged, see Fig. 11.3(e).

Now consider a CNOT (control-X) gate sandwiched between Hadamards as shown in Fig. 11.5(f). Consider the target (lower) qubit. If the control qubit does not act on it, the target qubit is just acted on by the two Hadamards which is equivalent to the identity, see Fig. 11.5(a). If the control qubit does act on the target qubit, the target qubit is acted on by the succession of gates $HXH$ which is equivalent to $Z$, see Fig. 11.5(b). Both these possibilities are taken care of by the equivalent circuit in Fig. 11.5(f)(i), which is control-Z gate. As illustrated in Fig. 11.5(e), the target and control qubits in a control-Z gate can be interchanged so Fig. 11.5(f)(i) is equivalent to Fig. 11.5(f)(ii). Now the target qubit is the upper one, and has the sequence of gates $H$ Ctrl-Z$H$ acting on it. Similar to the argument that showed Fig. 11.5(f) is equivalent to Fig. 11.5(f)(i), this is equivalent to Ctrl-X because of the identities in Fig. 11.5(a) and Fig. 11.5(c). Hence Fig. 11.5(f) is equivalent to Fig. 11.5(f)(iii).

So we see that a CNOT surrounded by Hadamards is equivalent to a CNOT gate without Hadamards but with the control and target qubits interchanged.

One could also derive this result by multiplying $4 \times 4$ matrices which is more tedious. However, for
11.B. DERIVATION OF SOME USEFUL IDENTITIES IN QUANTUM CIRCUITS

Figure 11.5: Some useful identities in quantum circuits.

completeness we will do it here. The CNOT gate has the matrix representation

\[
|00\rangle \quad |01\rangle \quad |10\rangle \quad |11\rangle \\
\begin{pmatrix}
1 & 0 & 0 & 0 \\
0 & 1 & 0 & 0 \\
0 & 0 & 0 & 1 \\
0 & 0 & 1 & 0
\end{pmatrix}.
\]

In this tensor product the control qubit is to the left. The target qubit (to the right) is flipped if the control qubit (to the left) is 1 (so, relative to the identity matrix, columns 3 and 4 are interchanged). In a CNOT gate with target and control qubits swapped, the left hand qubit is flipped if the right hand qubit is 1 (so columns 2 and 4 are interchanged). Hence we have

\[
|00\rangle \quad |01\rangle \quad |10\rangle \quad |11\rangle \\
\begin{pmatrix}
1 & 0 & 0 & 0 \\
0 & 0 & 0 & 1 \\
0 & 0 & 1 & 0 \\
0 & 1 & 0 & 0
\end{pmatrix}.
\]

The direct product \(\mathcal{H}^\otimes 2\) is given by

\[
\mathcal{H}^\otimes 2 = \frac{1}{\sqrt{2}} \begin{pmatrix}
H & H \\
H & -H
\end{pmatrix} = \begin{pmatrix}
1 & 1 & 1 & 1 \\
1 & -1 & 1 & -1 \\
1 & 1 & -1 & -1 \\
1 & -1 & -1 & 1
\end{pmatrix}.
\]
One can check by working out the matrix multiplication that

\[ U_{\text{CNOT,SWAP}} = H^\otimes 2 \, U_{\text{CNOT}} \, H^\otimes 2 , \]  

(11.19)
in agreement with Fig. 11.5(f). This is a bit tedious so I used Mathematica. It is more straightforward to use the circuit identities shown in Fig. 11.5.
Chapter 12

The Bernstein-Vazirani Algorithm

12.1 The Algorithm

Like the Deutsch algorithm, the Bernstein-Vazirani algorithm finds information about a black box function, but has a bigger speedup. It is very similar to the Deutsch-Josza algorithm which is set as a homework problem in the appendix at the end of this chapter.

Consider a function

\[ f(x) = a \cdot x \]  

(12.1)

where \( a \) and \( x \) have \( n \) bits while the function itself, \( f \), has one bit. The dot indicates a bitwise inner product with modulo 2 addition:

\[ a \cdot x \equiv a_0x_0 \oplus a_1x_1 \oplus \cdots \oplus a_{n-1}x_{n-1} \cdot \]  

(12.2)

The problem is to determine \( a \).

Let’s make sure that we understand the “dot”. For example for \( n = 4 \), if the bits of \( a \) are 1101 and the bits of \( x \) are 1110 (recall that the zeroth bit is the least significant, i.e. the rightmost one) then

\[ a \cdot x = (1 \times 0) + (0 \times 1) + (1 \times 1) + (1 \times 1) \mod 2 = 0 + 0 + 1 + 1 \mod 2 = 2 \mod 2 = 0. \]  

(12.3)

Hence, for these values of \( a \) and \( x \), \( f(x) = 0 \). If we take \( x = 1000 \) then \( f(x) = 0 + 0 + 0 + 1 \mod 2 = 1 \).

Classically we can only determine the bits of \( a \) one at a time. The \( k \)-th bit of \( a \) can be determined by feeding in \( x = 2^k \). To see this, consider the binary representations of \( a \) and \( x \):

\[ a = a_0 + a_12^1 + \cdots + a_k2^k + \cdots a_{n-1}2^{n-1}, \]  

\[ x = x_0 + x_12^1 + \cdots + x_k2^k + \cdots x_{n-1}2^{n-1}. \]  

(12.4)

Hence if \( x = 2^k \) then \( x_k = 1 \) while, for \( l \neq k \), \( x_l = 0 \), so \( a \cdot x = a_k \). Consequently \( f(2^k) = a_k \). We have to do this for each bit, \( k = 0, 1, 2, \cdots, n-1 \), so it requires \( n \) calls of the function.

The quantum algorithm succeeds in determining \( a \) with just one call!

A schematic diagram of a general reversible unitary transformation which takes an \( n \)-bit input and generates an \( m \)-bit output is shown in Fig. 10.2. For the Bernstein-Vazirani Algorithm there are \( n \) input qubits but only 1 output qubit. In addition, the unitary \( U_f \) is surrounded by Hadamards, as shown in Fig. 12.1. The upper register is set to \( |0\rangle_n \) and the lower qubit to \( |1\rangle \). This is the same circuit as for the Deutsch-Josza algorithm, see Appendix 12.A.

---

1One can either do the mod 2 operation after each addition or add up in the normal way and apply the mod 2 operation at the end. In either case, the result is 0 if an even number of terms in the sum are 1, and 1 if an odd number of terms are 1.
Figure 12.1: Circuit diagram for the Bernstein-Vazirani algorithm. In the final state the upper (input) register contains $|a\rangle$ while the lower (output) qubit reverts to its initial state $|1\rangle$. The desired value of $a$ can therefore be read off by measuring the upper register.

Acting with $H$ on $|0\rangle$ gives an equal linear superposition of the two basis states. Similarly acting with $H^\otimes n$ on $|0\rangle_n$ gives an equal superposition of the $2^n$ basis states. Hence, including the lower register, the state inputted to $U_f$ is

$$H^\otimes n |0\rangle_n \otimes H |1\rangle = \frac{1}{\sqrt{2^n}} \sum_{x=0}^{2^n-1} |x\rangle_n \otimes \frac{|0\rangle - |1\rangle}{\sqrt{2}} .$$

(12.5)

For each term in the superposition, the function $U_f$ acts in the same way as for the Deutsch algorithm described in Chapter 11. The lower qubit is flipped if $f(x) = 1$, which is the same as changing the sign of the state. This sign change is called phase kickback by Vathsan [Vat16]. If $f(x) = 0$ there is no change. Hence each term in the superposition acquires a factor of $(-1)^{f(x)}$, so the state of the system immediately the action of $U_f$ is

$$\frac{1}{\sqrt{2^n}} \sum_{x=0}^{2^n-1} (-1)^{f(x)} |x\rangle_n \otimes \frac{|0\rangle - |1\rangle}{\sqrt{2}} .$$

(12.6)

Next consider the effect of the Hadamards acting after $U_f$. The action on the lower qubit is to convert $(|0\rangle - |1\rangle)/\sqrt{2}$ to $|1\rangle$. However, the effect of $H^\otimes n$ acting on an arbitrary computational basis state $|x\rangle_n$ needs more thought. Consider first just one qubit. Then

$$H |x\rangle = \frac{1}{\sqrt{2}}(|0\rangle + (-1)^x|1\rangle) = \frac{1}{\sqrt{2}} \sum_{y=0}^{2^n-1} (-1)^{x\cdot y} |y\rangle .$$

(12.7)

Hence the effect of applying $H^\otimes n$ on an $n$-qubit computational basis state is

$$H^\otimes n |x\rangle_n = \sum_{y_{n-1}=0}^{1} \cdots \sum_{y_1=0}^{1} \sum_{y_0=0}^{1} (-1)^{\sum_{j=0}^{n-1} x_j y_j} |y_{n-1}\rangle \cdots |y_1\rangle |y_0\rangle ,$$

$$= \frac{1}{2^{n/2}} \sum_{y=0}^{2^n-1} (-1)^{x\cdot y} |y\rangle_n .$$

(12.8)

where $x \cdot y$ is the bitwise inner product with modulo 2 addition defined in Eq. (12.2), and we have used the fact that we only need to know whether $\sum_{j=0}^{n-1} x_j y_j$ is even or odd. Hence, combining Eqs. (12.6) and (12.8), the amplitude to find the upper register in state $|y\rangle_n \equiv |y_{n-1}\rangle \cdots |y_1\rangle |y_0\rangle$ is

$$c_y = \frac{1}{2^n} \sum_{x=0}^{2^n-1} (-1)^{f(x)+x\cdot y}$$

$$= \frac{1}{2^n} \prod_{j=0}^{n-1} \left[ \sum_{x_j=0}^{1} (-1)^{(a_j+y_j)x_j} \right] .$$

(12.9)
Let us evaluate this for the state where \( y_j = a_j \) for all \( j \). Then \((a_j + y_j)x_j = 2\) or \(0\) and, since this quantity is the power of \(-1\) we only need it modulo \(2\), so all the factors of \((-1)^{(a_j+y_j)x_j} \) are \(+1\), and consequently all \(2^n\) terms from the sums over the \(x_j\) add up in phase. The result is that the amplitude for \( y = a \) is \(1\). Since the total probability must add up to \(1\) this means that all the other amplitudes must be zero. To see that this is indeed the case, note that for each qubit where \( y_j \neq a_j \), \( a_j + y_j = 1 \) and so the sum over \(x_j\) for these qubits gives zero. The final result is a product over terms for each qubit and so we get zero, as required.

Including the output qubit, the final state is

\[ |a\rangle_n \otimes |1\rangle, \] (12.10)

and a measurement of the upper register in Fig. 12.1 gives \( a \), even though we made just one call to the function.

Since a classical computation of \( a \) requires \( n \) function calls, we have obtained a “quantum speedup” of \( n \). Note that the procedure is analogous to Deutsch’s algorithm. The first set of Hadamards generates a superposition of inputs to the gate \( U_f \) which “evaluates” the function for all \(2^n\) inputs using quantum parallelism, and then the second set of Hadamards destroys all the outputs apart from \( a \), using quantum interference.

### 12.2 An Alternative Derivation

Following Mermin [Mer07] and Vathsan [Vat16] it is useful to give an alternative derivation of how the circuit in Fig. 12.1 works, by giving an explicit construction of the black box \( U_f \). It is convenient to illustrate by a specific example. We take \( n = 5 \) and \( a = 11010 \) so \( x_0 = 0, x_1 = 1, x_2 = 0, x_3 = 1, x_4 = 1 \) (recall we read the bits from right, the least significant, to left, the most significant). The function \( a \cdot x \) can be implemented by the gates shown in Fig. 12.2.

![Figure 12.2: A circuit diagram for \( n = 5 \) to implement the function \( f(x) = a \cdot x \) with \( a = 11010 \), i.e. \( f(x) = x_1 + x_3 + x_4 \) mod 2. The circuit flips the output qubit, the lowest one, initialized to \( y \), whenever \( x_1 \) or \( x_3 \) or \( x_4 \) is 1. (Note that flipping \( y \) is equivalent to adding 1 to \( y \) mod 2.) Hence the final value of the output qubit is \( y \oplus (a \cdot x) \) as required.](image)

To incorporate \( U_f \) into the Bernstein-Vazirani algorithm, we sandwich it in between Hadamards, see Fig. 12.1, and note that the Hadamards interchange control and target qubits in the CNOT (control-X) gates, see Fig. 11.5(f) in Chapter 11. As before, the initial upper register is \( |0\rangle_n \) and the lower register is \( |1\rangle \). We see immediately from Fig. 12.3 that \( a \) is directly imprinted in the final state of the input register. There does not appear to be any parallelism and interference.

---

\(^2\)To understand the reason for the quotation marks see the discussion at the end of Sec. 10.2.
CHAPTER 12. THE BERNSTEIN-VAZIRANI ALGORITHM

Figure 12.3: Sandwiching the circuit for $U_f$ in Fig. 12.2 between Hadamards, and realizing that the effect of the Hadamards is to interchange the control and target qubits in the CNOT (control-$X$) gates, we see immediately that the final state of the input register contains $a = 11010$.

Hence these two explanations of the Bernstein-Vazirani algorithm are quite different. To quote Mermin [Mer07]:

"The first applies $U_f$ to the quantum superposition of all possible inputs and then applies operations which leads to perfect destructive interference of all states in the superposition except for the one in which the upper (input) register is in the state $|a\rangle$. The second suggests a specific mechanism for representing the subroutine that executes $U_f$ and then shows that sandwiching such a mechanism between Hadamards automatically (my italics) imprints $a$ on the upper register. Interestingly, quantum mechanics appears in the second method only because it allows the reversal of the control and target qubits of a cNOT operation solely by means of 1-qubit (Hadamard) gates."

(I have used the conventional spelling of "qubit" rather than Mermin’s idiosyncratic "Qbit").

Appendices

12.A Homework Problem on the Deutsch-Josza Algorithm

This is an extension of the Deutsch algorithm discussed in class. Recall that in Deutsch’s algorithm the input is one bit and the output is also one bit. In the Deutsch-Josza algorithm, the output is still one bit but the input has $n$ bits, so there are $2^n$ distinct inputs. We are told that either the function is "constant" (in which case the function outputs the same value for all $2^n$ inputs) or is "balanced" (in which case an equal number of inputs give the results 1 and 0). Clearly this is a very artificially constructed problem.

The circuit for the Deutsch-Josza algorithm, shown in Fig. 12.4, is the same as for the Bernstein-Vazirani algorithm.

The function $U_f$ acts as follows on computational basis states $|x\rangle_n$ and $|z\rangle$:

$$U_f|x\rangle_n|z\rangle = |x\rangle_n|z \oplus f(x)\rangle,$$

where $x$ is an $n$-bit integer, $|x\rangle$ is the state of the $n$-qubit upper register in the figure, $z$ and $f(x)$ are 1-bit integers, and $|z\rangle$ is the lower qubit in the figure. The lower qubit is initialized to $|1\rangle$, and the upper register is initialized to $|0\rangle_n$. 

1. Show that

\[ |\psi_0\rangle_n = H^\otimes n |0\rangle_n = \frac{1}{\sqrt{2^n}} \sum_{x=0}^{2^n-1} |x\rangle_n, \]  

so the input to the function \( U_f \) is the uniform superposition of all \( 2^n \) basis states.

2. Show that after the action of \( U_f \) the state of the upper register is

\[ |\psi_1\rangle_n = \frac{1}{\sqrt{2^n}} \sum_{x=0}^{2^n-1} (-1)^{f(x)} |x\rangle_n. \]  

3. Show that after the action of the second set of Hadamards on the \( n \)-qubit register, the state of that register is

\[ |\psi_2\rangle_n = H^\otimes n |\psi_1\rangle_n = \frac{1}{2^n} \sum_{x,y=0}^{2^n-1} (-1)^{f(x)+x\cdot y} |y\rangle_n, \]  

where \( x \cdot y \) is the bitwise inner product of \( x \) and \( y \) with modulo 2 addition:

\[ x \cdot y = x_0 y_0 \oplus x_1 y_1 \oplus \ldots \oplus x_{n-1} y_{n-1}. \]

4. The upper register is then measured, and an \( n \)-bit integer \( y \) is obtained. Show that if the function is a constant then \( y = 0 \) with probability 1. Show also that if the function is balanced then one must get a non-zero value of \( y \). Hence the Deutsch-Josza algorithm succeeds with just one function call.

5. How does this compare with a classical approach? The only thing one can do classically is keep computing \( f(x) \) for different values of \( x \) and seeing if one gets more than one value for the output. If the function is balanced, one would probably get different outputs quite quickly. If the function is constant one would need to evaluate half the inputs (plus 1), i.e. \( 2^{n/2} + 1 \), to be 100% sure that the function is not balanced. This is exponentially (in \( n \)) worse than the quantum algorithm. However, this is arguably not fair. We may well be content to establish that the function is constant with some high probability, a bit less than one. If the function is constant, how many function calls would you need classically to rule out the possibility that it is balanced with a probability of error of no more than (i) \( 10^{-3} \) and (ii) \( 10^{-6} \).

Note: For simplicity, assume that the number of function calls is much less than \( 2^{n/2} \), the number of values of \( x \) which give the same result if the function is balanced.

\( ^{3} \)For later quantum algorithms we will only be able to solve the problem with high probability. Since we have to give up 100% certainty in the quantum case, we should not insist on 100% certainty here from the classical algorithm.
12.B  Homework problem on making a Toffoli Gate out of 1-qubit and 2-qubit gates

We noted in Sec. 8.1 that for classical reversible computation we need three-bit gates, such as the Toffoli gate, in addition to 1-bit and 2-bit gates, to be able to perform universal computation. However, three qubit gates are, fortunately, not needed in quantum computation because the necessary three-bit gates can be constructed out of 1-qubit and 2-qubit gates.

Here we consider the quantum Toffoli gate, which is a control-control-NOT (C-C-NOT) gate. The target qubit $z$ is flipped if both the control qubits, $x$ and $y$, are 1 and is otherwise unchanged.

We present here the method of constructing the Toffoli gate out of 1-qubit and 2-qubit gates as a homework problem (including help).

1. Consider the following circuit for an arbitrary unitary operator $V$:

   ![Circuit Diagram]

   Show that it acts with $V^2$ on $|z\rangle$ if both $x$ and $y$ are 1 and otherwise does nothing.

   **Hint:** One possible way of approaching this question (though not the only way and not the most elegant way) is to consider separately what happens for the four possible input values of the control qubits $x$ and $y$, 00, 01, 10, and 11.

2. Now take $V$ to be the following 1-qubit gate:

   \[
   V = (1 - i) \frac{(1 + iX)}{2}.
   \]  

   Show that $V^\dagger V = \mathbb{1}$, and hence $V$ is unitary. Show also that $V^2 = X$ and hence the above circuit is a quantum Toffoli gate.

**Note:**

- One sometimes says that $V$ is the “square root of $X$”.
- Unlike other unitary operators considered in this course, $V$ is not its own inverse.
- Vathsan [Vat16] discusses how to construct a general Controlled-$V$ gate in her Sec. 7.4.1.
Chapter 13

Simon’s Algorithm

So far we have studied Deutsch’s algorithm in Chapter 11 which gave a quantum speedup of a factor of 2, and the Bernstein-Vazirani algorithm in Chapter 12 which gave a speedup of \( n \), where \( n \) is the size of the problem. Next we consider a problem, due to Daniel Simon, which gives an exponential speedup in \( n \). Like the previous algorithms it has an artificial character and is not of practical use, but it has features in common with the vastly more useful algorithm of Shor for factoring integers which we shall spend quite some time on next. Like Shor’s algorithm it is of a probabilistic nature.

In Simon’s problem we are given a black box function which takes an \( n \)-bit input and has the property that

\[
f(x \oplus a) = f(x),
\]

(13.1)

where \( a \) is a non-zero \( n \)-bit integer and \( \oplus \) means bitwise addition modulo 2. Adding \( a \) twice to \( x \) (modulo 2) gives back \( x \), i.e.

\[
x + a + a = x
\]

(13.2)

since adding a bit to itself gives 0 (mod 2) irrespective of whether that bit is 0 or 1. Hence

\[
f(x) = f(x + a) = f(x + a + a)
\]

(13.3)

and so on, so \( f(x) \) is periodic under bitwise mod 2 addition. We are told that for every \( x \) there is only one other input to the function, \( x \oplus a \), which gives the same output, so there are \( 2^{n-1} \) distinct values of \( f \). Hence we assume that we can represent \( f \) by \( n - 1 \) qubits.

If we input different values of \( x \) and find a repeated output, i.e. if \( f(x_i) = f(x_j) \), then \( x_j = x_i \oplus a \). If we add \( x_i \) to both sides (bitwise addition modulo 2) we get

\[
a = x_i \oplus x_j.
\]

(13.4)

so we obtain \( a \) if we can find two values of \( x \) which give the same function value.

<table>
<thead>
<tr>
<th>( x )</th>
<th>000</th>
<th>001</th>
<th>010</th>
<th>011</th>
<th>100</th>
<th>101</th>
<th>110</th>
<th>111</th>
</tr>
</thead>
<tbody>
<tr>
<td>( f(x) )</td>
<td>3</td>
<td>2</td>
<td>2</td>
<td>3</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>0</td>
</tr>
</tbody>
</table>

Table 13.1: An example with \( n = 3 \) bits of the type of function that is considered in Simon’s algorithm. The function satisfies \( f(x) = f(x \oplus a) \) for some non-zero \( a \). To determine \( a \) we look for repetitions. The first occurs for \( f(1) = f(2) = 2 \). Hence, according to Eq. (13.4), \( a = 001 \oplus 010 = 011 = 3 \). The other repetitions satisfy this same condition as you can check.

The problem is to determine \( a \) with as few calls to the function as possible. An example is shown in Table 13.1.
Classically this problem is hard, by which we mean that number of function calls grows exponentially with \( n \). All one can do is call the function with successive values of \( x \) until one finds a repeated output, i.e. \( f(x_i) = f(x_j) \), which gives us \( a \) from Eq. (13.3). After \( m \) calls to the function we have compared \( m(m-1)/2 \) pairs. For a reasonable chance of success we need \( \frac{1}{2}m(m-1) \sim 2^n \), so \( m = O(2^n/2) \), i.e. exponential in the number of bits \( n \).

The circuit to solve this problem quantum mechanically is similar to that in the Bernstein-Vazirani algorithm except that the lower register has enough qubits to contain the function values, i.e. \( n-1 \). Also the phase kickback is not used, so the lower register is initialized to \( |0\rangle^{n-1} \) rather than \( |1\rangle \) and we do not have Hadamards on the lower register. A final difference is that we measure first on the lower register rather than the upper one. The circuit diagram is shown in Fig. 13.1.

After the first Hadamards in the upper register the state of the system is

\[
|\psi_0\rangle = \frac{1}{2^{n/2}} \sum_{x=0}^{2^n-1} |x\rangle_n \otimes |0\rangle_{n-1}.
\]  

(13.5)

The function call makes the transformation \( |x\rangle_n \otimes |y\rangle_{n-1} \rightarrow |x\rangle_n \otimes |y \oplus f(x)\rangle_{n-1} \), see Fig. 12.1 in Chapter 12. Here \( y = 0 \) so, after the function call the state becomes

\[
|\psi_1\rangle = \frac{1}{2^{n/2}} \sum_{x=0}^{2^n-1} |x\rangle_n \otimes |f(x)\rangle_{n-1}.
\]  

(13.6)

A measurement is then done on the lower register which will record some value of the function, \( f_0 \) say, for which there are two values of \( x \) which we denote by \( x_0 \) and \( x_0 \oplus a \). Hence, immediately after the measurement, the state of the system is

\[
|\psi_2\rangle = \frac{|x_0\rangle_n + |x_0 \oplus a\rangle_n}{\sqrt{2}} \otimes |f(x_0)\rangle_{n-1}.
\]  

(13.7)

If we were now to measure the upper register, we would get either \( x_0 \) or \( x_0 \oplus a \). At first glance, this might seem like progress since we appear to be halfway there. If we could just get the other number, we would have \( a \). However there is no way to get both. If we could clone the state several times and measure each clone then, with high probability, we would be able to determine both of them. However, the no-cloning theorem says that we can’t clone an arbitrary, unknown state. Also, repeating the whole procedure doesn’t help because, with high probability, we would get a different function value, \( ˜f_0 \), and hence a different pair \( ˜x_0 \) and \( ˜x_0 \oplus a \), from which again we would not be able to extract \( a \).
As in Deutsch’s algorithm and the Bernstein-Vazirani algorithm, we must do some processing \textit{before} the final measurement. As we showed in Chapter 12 on the Bernstein-Vazirani algorithm, the effect of Hadamards on each qubit of an $n$-qubit register is given by
\[ H^\otimes n |x\rangle_n = \frac{1}{2^{n/2}} \sum_{y=0}^{2^n-1} (-1)^{x \cdot y} |y\rangle_n , \] 
where $x \cdot y$ is the bitwise inner product modulo 2,
\[ x \cdot y \equiv x_0y_0 \oplus x_1y_1 \oplus \cdots \oplus x_{n-1}y_{n-1} \mod 2 . \] 
Hence, applying Hadamards to the $n$-qubit upper register in state $|\psi_2\rangle$ in Eq. (13.7), the state of that register becomes
\[ |\psi_3\rangle_n = \frac{1}{\sqrt{2}} \frac{1}{2^{n/2}} \sum_{y=0}^{2^n-1} \left[ (-1)^{x_0 \cdot y} + (-1)^{(x_0 \oplus a) \cdot y} \right] |y\rangle_n . \] 
Now $\left( x_0 \oplus a \right) \cdot y = (x_0 \cdot y) \oplus (a \cdot y)$ and we note that $a \cdot y = 0$ or 1. If $a \cdot y = 1$ then $(-1)^{(x_0 \oplus a) \cdot y} = (-1)^{x_0 \cdot y} (-1)^a y = -(1)^{x_0 \cdot y}$, so the two terms in Eq. (13.10) cancel. Hence the only terms with a non-zero amplitude are those with $a \cdot y = 0$.

A measurement on the upper register then gives, with equal probability, one value of $y$ with $a \cdot y = 0$. This is a linear equation for the $a_i$, the bits of $a$. If we can find $n$ linearly-independent equations for the $a_i$, we can obtain the solution. Hence we have to repeat the procedure, each time getting a value for $f_0$ and $y$. As discussed in Appendix G of Mermin [Mer07] one needs to run the algorithm \textit{a little more} than $n$ times because the equations one gets each time for the $a_i$ are not necessarily linearly independent. The result is that if one runs $n + x$ times, then the probability of getting $n$ linearly independent equations (and hence the solution for the $a_i$) is is greater than
\[ 1 - \frac{1}{2^x + 1} . \] 
Hence there is less than one chance in a million of failure if one calls the function $n + 20$ times. A crucial point in this expression is that the number of calls beyond $n$ needed to find a solution with some high probability does not depend on $n$.

The occurrence of probability, and some arcane mathematical arguments to prove that one does get the solution with high probability within the specified number of runs, is characteristic of several quantum algorithms including Shor’s.

In the case of Simon’s problem, the classical algorithm takes of order $2^{n/2}$ function calls whereas the quantum algorithm finds the answer with high probability with little more than $n$ calls.\footnote{This is the mod 2 version of the usual distributive rule for addition and multiplication: $a \times (b + c) = (a \times b) + (a \times c)$.} This is an \textit{exponential speedup}.\footnote{In the interests of full disclosure I should state that one also needs to solve $n$ linear equations on a classical computer, which takes of order $n^3$ steps. A algorithm which takes a time proportional to a power of the problem size $n$ is said to be \textit{polynomial}. Since classical hardware is cheap it is not clear if one should include this time on the classical computer in the computational cost of Simon’s algorithm. However, since $n^3$ is polynomial, even if one does include this time the comparison is still between a polynomial quantum (+classical) algorithm and an exponential purely classical algorithm, which is still an exponential speedup, see footnote 3.}

Finally a few words of anticipation for Shor’s algorithm which we will do next. Simon’s problem considers a function which is periodic under bitwise modulo 2 addition, see Eq. (13.3). Shor’s algorithm

\footnote{An algorithm which takes a time proportional to a power of the problem size is said to have \textit{polynomial complexity}, while if the time increases exponentially with size (or exponentially with a power of the size) it is said to have \textit{exponential complexity}. If one algorithm has polynomial complexity and another has exponential complexity then the former is said to have an exponential speedup compared with the latter.}
investigates functions which are periodic under *ordinary* addition: \( f(x + a) = f(x) \), which is much more useful. In Simon’s problem, the action of the \( n \)-Hadamards in Eq. (13.8) can be written

\[
H_{\otimes n}|x\rangle_n = \frac{1}{2^{n/2}} \sum_{y=0}^{2^n-1} e^{i\pi x \cdot y} |y\rangle_n ,
\] (13.12)

Since \( x \cdot y \) is the bitwise inner product modulo 2, it only takes values 0 and 1, so the phases in the complex exponential are just 0 and \( \pi \). The core of Shor’s algorithm is a quantum Fourier transform (QFT), where an essential difference from Eq. (13.12) is that the bitwise inner product is replaced by ordinary multiplication. Hence the QFT generates many different phases, with the result that, unlike Simon’s algorithm, it cannot, in general, be constructed entirely out of 1-qubit gates. Fortunately, it can be constructed entirely out of 1- and 2-qubit gates. All this and more will be discussed in Chapter 18.
Chapter 14

Factoring and RSA
(Rivest-Shamir-Adleman) Encryption

Shor’s famous quantum algorithm, to be discussed in detail in Chapter 18, factors large integers much more efficiently than any known classical algorithm. Factoring is not just of interest to mathematicians, however, because the difficulty of factoring is at the heart of the popular RSA method of encrypting sensitive information sent via the internet (or some other public channel). While RSA is not the only method used to encrypt information, my understanding is that some version of Shor’s algorithm can be used to crack other encryption methods such as Diffie-Hellman. RSA stands for the names of its inventors, Rivest, Shamir and Adleman.

This chapter is a \LaTeX{} copy of a Mathematica notebook, the original of which is available at https://young.physics.ucsc.edu/150/rsa.nb In it, the RSA algorithm is implemented, parameters are chosen, and random messages are generated. These are encrypted, the encrypted messages are decrypted, and a check is made that the original message is recovered. If you have Mathematica you can run the notebook version and verify that the RSA algorithm works.

Suppose that Bob wants to receive a message from Alice on the internet (a public channel). Anything sent on a public channel can be intercepted by others. How can Bob and Alice agree on a coding scheme and then send each other coded messages which can be decoded by the other person but not by anyone “sniffing” on the internet? This has to be accomplished by only sending messages down the public channel.

We will now describe the RSA encryption scheme for doing this. It uses a result of number theory which we will quote but not prove. To receive the message from Alice, Bob picks two large prime numbers \(p\) and \(q\), and sends to Alice, on the public channel, their product

\[N = pq,\]

but not \(p\) and \(q\) separately. \(N\) is taken to be large enough, typically a few thousand bits, that it cannot be factored on a classical computer. You might ask how can one choose the large prime numbers \(p\) and \(q\). If one selects a large integer \(N\) at random it can be shown that the probability that it is prime is about \(1/\ln N\). Hence, even if \(N\) has, say, 400 digits (around 1000 bits) you only have to test a few hundred to a thousand random integers to typically find a prime number. But can one efficiently test if a number is prime? It turns out that one can, even though, if the number is found to be not prime, there is no known efficient classical algorithm to determine the prime factors. The website http://mathworld.wolfram.com/PrimalityTest.html explains how the test for primality is done in Mathematica.

Bob also sends a large “encoding number” \(c\) which has no factors in common with \((p - 1)(q - 1)\). If there are no factors in common then the greatest common divisor (GCD) is 1. The GCD of two
integers is easily determined by Euclid’s algorithm discussed in Sec. 14.A. According to Appendix J of Mermin [Mer07], the probability that two large random integers have no common factors is greater than $1/2$, so it is not difficult to find a suitable value for $c$. Hence the public key (available to everyone) is $N$ and $c$. Since Bob knows both $p$ and $q$, and hence $(p−1)(q−1)$, he can also determine the integer $d$ such that

$$cd = 1 \pmod{(p−1)(q−1)}.$$ (14.2)

Let us remind ourselves of this mod function. The value of $a \pmod{b}$ is the result after one subtracts (or adds) the appropriate multiple of $b$ to $a$ to get a value which lies in the range $0$ to $b−1$. If $a$ is positive, things are simple, one subtracts a multiple of $b$ (possibly 0) so the mod function is just the remainder after integer division. Hence, for example, $9 \pmod{5} = 4$ because $9/5 = 1$ remainder 4. If $a$ is negative one has to add a multiple of $b$, so, for example, $(-13) \pmod{5} = 2$ (since $-13 + (3 \times 5) = 2$).

The above equation, $cd = 1 \pmod{something}$, looks strange at first. If $c$ is an integer we normally think that its inverse should be a fraction. However, here $d$ is also an integer, and the product of two integers can give 1 if we use modular arithmetic. For example if $c = 5$ and $d = 3$ then $cd = 15$, and $cd \pmod{7} = 1$ (since $15 = (7 \times 2) + 1$).

The algorithm for computing $d$ in Eq. (14.2) is efficient and an extension of Euclid’s algorithm. It is given in Sec. 14.B and in Appendix J of Mermin [Mer07]. It turns out that $d$ is unique. Hence Alice, and anyone else sniffing on the public channel, knows $N$ and $c$ (but not $p$, and $q$, and hence not $d$). The private key (known only to Bob) is $p$ and $q$ (and hence $d$).

Alice breaks up her message into chunks of each containing a number of bits less than the number of bits of the integer $N$. Each chunk is then a binary number less than $N$. Let’s denote by $a$ the numerical value of one chunk.

$a$ is the original message.

Using the values of $N$ and $c$ that Bob has sent, she computes

$$b = a^c \pmod{N} \text{ the encoded message.}$$ (14.3)

The encoded message $b$ is another large integer, and is sent down the public channel from Alice to Bob.

Bob knows not only $c$ and $N$, but also the value of $d$. Here number theory kicks in and shows that the original (unencoded) message $a$ is given by

$$a = b^d \pmod{N} \text{ (the original message is recovered).}$$ (14.4)

For a proof of this result see the book by Mermin [Mer07]. Bob can compute it because he knows $d$, but anyone sniffing on the public channel does not. However, if a third person, traditionally called Eve, listening on the public channel, could factor $N$ (which is sent down the public channel) into its factors $p$ and $q$, she would then have $(p−1)(q−1)$ and, since $c$ is also sent down the public channel, she could determine $d$ where $cd = 1 \pmod{(p−1)(q−1)}$ using the algorithm given at the end of this chapter and in Mermin’s Appendix J. Hence she could find the original unencrypted message $a$ from Eq. (14.4).

Let’s do a simple example. We will take

$$p = 7, \quad q = 13, \quad \text{so } N = 91.$$ (14.5)

For the encoding integer we take $c = 11$, which has no factors in common with $(p−1)(q−1) = 6 \times 12 = 72$. As shown in Sec. 14.B using the extended Euclid algorithm one finds that $d = 59$. (Let’s verify this: $11 \times 59 = 649 = (9 \times 72) + 1$.) The Mathematica code below sets these values, checks that $p$ and $q$ are prime while $N$ is not, and that $cd = 1 \pmod{(p−1)(q−1)}$. (Note: in Mathematica commands I use $n$ rather than $N$ because $N$ has a special meaning in Mathematica.) The code then
generates a message $a$ by computing a random integer between 0 and $N - 1$, and next computes the encoded message $b$ from $b = a^c \pmod{N}$. It then computes $b^d \pmod{N}$ and checks that it gives back the original message $a$. If you have Mathematica you can run the code several times (each time a different random value for the message $a$ will be generated) and see that the original message is always returned.

In[1]:= p=7; q=13; c=11; d=59; n=p*q
Out[1]= 91

We check that $p$ and $q$ are prime. The Mathematica command PrimeQ[p] returns “True” if $p$ is prime and “False” if it is not.

In[2]:= PrimeQ[p]
In[3]:= PrimeQ[q]
In[4]:= PrimeQ[n]

We check that $cd = 1 \pmod{(p-1)(q-1)}$.

In[5]:= Mod[c * d, (p-1)(q-1)]
Out[5]= 1

We generate a random message.

In[6]:= mess = Random[Integer, n - 1]
Out[6]= 51

We compute the encoded message.

In[7]:= encodedmess = Mod[mess^c, n]
Out[7]= 25

We decode the encoded message and check that we recover the original message.

In[8]:= recoveredmess = Mod[encodedmess^d, n]
Out[8]= 51
In[9]:= recoveredmess == mess
Out[9]= True

Hence the message was successfully decoded.

### Appendices

#### 14.A The Euclidean Algorithm

We want to efficiently find the Greatest Common Divisor (GCD) of two integers. This is the largest factor that they have in common. As a simple example, the GCD of 24 and 9 is 3.

Suppose we want the GCD of two numbers $a_0$ and $b_0$ with $a_0 > b_0$. We proceed iteratively. At each stage, the new value of $a$ is equal to the old value of $b$, and the new value of $b$ is equal to the remainder when the old value of $a$ is divided by the old value of $b$, i.e.

\[
\begin{align*}
  a_{n+1} &= b_n \\
  b_{n+1} &= a_n - \left[ a_n/b_n \right] b_n
\end{align*}
\]  

(14.6) which is the same as $b_{n+1} = a_n \mod b_n$. 

CHAPTER 14. FACTORING AND RSA (RIVEST-SHAMIR-ADLEMAN) ENCRYPTION

where \([\cdots]\) means the integer part of the quantity in brackets. \(a_n\) and \(b_n\) decrease at successive iterations and maintain the inequality \(a_n > b_n\). Also \(a_n\) and \(b_n\) have the same common factors as \(a_0\) and \(b_0\). Eventually we get to a stage where \(b_{n+1} = 0\). This means that \(a_n\) is divisible by \(b_n\) so \(b_n\) is the greatest common divisor.

As an example we take \(a_0 = 24, b_0 = 9\),

<table>
<thead>
<tr>
<th>(n)</th>
<th>(a_n)</th>
<th>(b_n)</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>24</td>
<td>9</td>
</tr>
<tr>
<td>1</td>
<td>9</td>
<td>6</td>
</tr>
<tr>
<td>2</td>
<td>6</td>
<td>3</td>
</tr>
<tr>
<td>3</td>
<td>3</td>
<td>0</td>
</tr>
</tbody>
</table>

Hence the GCD of 24 and 9 is \(b_2 (= 3)\).

14.B Extension of the Euclidean Algorithm to find an inverse modulo an integer

Given \(a\) and \(c\) which have no common factors, and \(a < c\), we want to find \(d\) where

\[
    cd = 1 \mod a. \tag{14.7}
\]

The greatest common divisor of \(c\) and \(a\) is 1 since, by assumption, they have no common factors. We go through the Euclid algorithm

\[
    a_{n+1} = c_n
    
    c_{n+1} = a_n - \left\lfloor \frac{a_n}{c_n} \right\rfloor c_n
\]

until we get to the stage where \(c_n = 1\), the greatest common divisor. One can then obtain \(d\) by working backwards through the iterations. This is best shown by an example. We take \(p = 7, q = 13\), as in example above, so we have \(a = (p-1)(q-1) = 72\) and hence we initialize \(a_0 = 72\). We also take \(c = 11\) (again as in the example) which has no factors in common with \(a\), and so initialize \(c_0 = 11\). Hence the Euclid algorithm proceeds as follows

<table>
<thead>
<tr>
<th>(n)</th>
<th>(a_n)</th>
<th>(c_n)</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>72</td>
<td>11</td>
</tr>
<tr>
<td>1</td>
<td>11</td>
<td>6</td>
</tr>
<tr>
<td>2</td>
<td>6</td>
<td>5</td>
</tr>
<tr>
<td>3</td>
<td>5</td>
<td>1</td>
</tr>
</tbody>
</table>

\(a_0 = a, c_0 = c\) (the initial values)

\(a_1 = c_0, c_1 = a_0 - 6c_0 = 6\)

\(a_2 = c_1, c_2 = a_1 - c_1 = 5\)

\(a_3 = c_2, c_3 = a_2 - c_2 = 1\) (\(c_3 = 1\) so we stop).

Hence working backwards,

\[
    1 = a_2 - c_2 = c_1 - (a_1 - c_1) = 2c_1 - a_1 = 2(a_0 - 6c_0) - c_0 = 2a_0 - 13c_0 (= 2a - 13c). \tag{14.9}
\]

We want to take this \((\mod a)\). Now \(2a \ (\mod a) = 0\). Also \(-13c\) is negative which is inconvenient. However, we can add to it \(ac\) (which is zero \((\mod a)\)). Hence

\[
    1 = 2a - 13c \ (\mod a) = -13c \ (\mod a) = (-13 + a)c \ (\mod a) = 59c(\mod a), \tag{14.10}
\]

where we used that \(a = 72\) to get the last equality. Hence \(d = 59\) as stated in the above example.
In this chapter, we explain how finding the period of a certain function will enable us to factor integers. We will also illustrate the technique with a simple example. This will probably seem a strange approach for factoring, and is not the preferred method on a classical computer, but it is the method used by Shor in his quantum algorithm.

We take two large primes \( p \) and \( q \) and form the product 
\[
N = pq .
\] (15.1)

The goal is to find the factors \( p \) and \( q \) given only the product \( N \). This is a problem which is hard classically. For applications in cryptography \( p \) and \( q \) may have around 600 digits (around 2000 bits).

We proceed by choosing a random integer \( a \) which has no factors in common with \( N \). Whether or not \( a \) and \( N \) have a common factor can be determined efficiently using Euclid’s algorithm, which was described in Sec. 14.A. In the very unlikely event that \( a \) and \( N \) do have a common factor we have found a factor of \( N \) and the problem is solved. Otherwise we compute the following function
\[
f(x) \equiv a^x \pmod{N}
\] (15.2)

for \( x = 1, 2, \ldots \). As stated, \( a \) and \( N \) have no common factors, and for this case one can show that eventually we will get \( f(x) = 1 \) for some value, \( x = r \) say, so
\[
a^r \pmod{N} \equiv 1.
\] (15.3)

The function then repeats since
\[
f(x + r) \equiv a^x \equiv a^{x+r} \pmod{N} \equiv a^x \pmod{N} \equiv a^r \pmod{N} = f(x),
\] (15.4)

using Eq. 15.3. Hence \( r \) is the period of the function.

We we illustrate with a simple example,
\[
N = pq = 91, \quad \text{with factors } p = 13, q = 7.
\] (15.5)

We also take \( a = 4 \), which has no factors in common with 91. We plot \( f(x) \equiv 4^x \pmod{91} \) in Fig. 15.1. The periodic nature is clear, and the period is found to equal 6 by inspection. Let’s make sure we
CHAPTER 15. USING PERIOD FINDING TO FACTOR AN INTEGER

Figure 15.1: The function $f(x) \equiv 4^x \pmod{91}$. The period is seen by inspection to equal 6.

Understand how this figure is obtained by working out the values of $4^x \pmod{91}$ for $x = 1, 2, \ldots, 6$.

- $x = 1, \quad a^x = 4, \quad (15.6a)$
- $x = 2, \quad a^x = 16, \quad (15.6b)$
- $x = 3, \quad a^x = 64, \quad (15.6c)$
- $x = 4, \quad a^x \equiv 64 \times 4 = 256 \equiv 2 \times 91 + 74 \equiv 74 \pmod{91}, \quad (15.6d)$
- $x = 5, \quad a^x \equiv 74 \times 4 = 296 \equiv 3 \times 91 + 23 \equiv 23 \pmod{91}, \quad (15.6e)$
- $x = 6, \quad a^x \equiv 23 \times 4 \equiv 91 + 1 \equiv 1 \pmod{91}. \quad (15.6f)$

The plot in Fig. 15.1 seems to have a fairly regular behavior, but such smooth behavior is exceptional and occurs here only because of the particular choice of parameters. Figure 15.2 shows a plot for the same value of $N$ but with $a = 19$. This is a much more random looking figure, as is typical. In this case the period is $r = 12$. The apparently random shape of $f(x)$ means that one cannot estimate the period by taking a few nearby values of $x$ and extrapolating.

We now need to be lucky in two respects:

- The period $r$ must be even. This means that $r/2$ is an integer and so is $a^{r/2}$. Hence we can write
  $$0 \equiv a^r - 1 \equiv (a^{r/2} - 1)(a^{r/2} + 1) \pmod{pq}. \quad (15.7)$$

- We need that
  $$a^{r/2} + 1 \not\equiv 0 \pmod{pq}. \quad (15.8)$$

It is automatically true that $a^{r/2} - 1 \not\equiv 0 \pmod{pq}$ because $x = r$ is the smallest power for which $a^x - 1 \equiv 0 \pmod{pq}$. Hence, if Eq. (15.8) is true, neither $a^{r/2} + 1$ nor $a^{r/2} - 1$ is divisible by $N = pq$ but, according to Eq. (15.7), their product is. Since $p$ and $q$ are primes, this is only
possible if $a^{r/2} + 1$ is a multiple of one of the factors, $p$ say, i.e. $a^{r/2} + 1 = Cp$, and $a^{r/2} - 1$ is a multiple of the other one $q$, i.e. $a^{r/2} - 1 = C'q$ ($C$ and $C'$ are constants). Consequently $p$ is the greatest common divisor of $N (= pq)$ and $a^{r/2} + 1 (= Cp)$, and $q$ is the greatest common divisor of $N$ and $a^{r/2} - 1$. We can therefore find $p$ and $q$ using the Euclidean algorithm mentioned earlier.

What are the odds that we will be doubly lucky in this way. According to Appendix M in Mermin [Mer07] the probability is greater than 0.5 for large $N$. If one is unlucky one tries a different choice for $a$. Since the probability of success is quite high at each attempt, one does not have to repeat the process very many times to succeed with very high probability.

Back to our example. For $N = 91, a = 4$ we found $r = 6$. Indeed we are lucky! This is even. Also $a^{r/2} + 1 = 65 \not\equiv 0 \pmod{91}$. So we are doubly lucky! However, this is not remarkable. As noted above the probability of this double luck is greater than 0.5 (at least for large $N$).

Hence one of the factors is the greatest common divisor of 91 and $a^{r/2} + 1 = 65$. The other factor is the greatest common divisor of 91 and $a^{r/2} - 1 = 63$.

Applying Euclid’s algorithm, described in Sec. 14.A to $f_0 = 91, g_0 = 65$:

$$
\begin{align*}
    f_1 &= 65, \\
    g_1 &= 91 - \lfloor 91/65 \rfloor 65 = 91 - 65 = 26, \\
    f_2 &= 26, \\
    g_2 &= 65 - \lfloor 65/26 \rfloor 26 = 65 - 52 = 13, \\
    f_3 &= 13, \\
    g_3 &= 26 - \lfloor 26/13 \rfloor 13 = 26 - 26 = 0.
\end{align*}
$$

Hence the GCD is $g_2 = 13$, which is indeed one of the factors of 91. By the same process the GCD of 63 and 91 is found to be 7, the other factor of 91.

---

1So $(a^{r/2} + 1)(a^{r/2} - 1) = CC'pq = CC'N$ which shows that $(a^{r/2} + 1)(a^{r/2} - 1)$ is a multiple of $N$. However, neither $a^{r/2} + 1$ nor $a^{r/2} - 1$ separately are a multiple of $N$ since this is excluded, at least if $a^{r/2} + 1 \not\equiv 0 \pmod{pq}$ as assumed.
Chapter 16

The Fourier Transform and the Fast Fourier Transform (FFT)

16.1 Introduction

The standard Fourier Transform concerns a continuous function, $x(t)$ say. For descriptive purposes it will be convenient to think of $t$ as time, but this is not essential. In the Fourier transform we decompose $x(t)$ into its components at different “frequencies” $y(\omega)$ as follows:

$$y(\omega) = \frac{1}{\sqrt{2\pi}} \int_{-\infty}^{\infty} e^{i\omega t} x(t) \, dx.$$  
(16.1)

If $x(t)$ comprises oscillations at frequencies at and around $\omega_0$, say, then $y(\omega)$ will be peaked at $\omega = \omega_0$.

There is also an inverse Fourier transform,

$$x(t) = \frac{1}{\sqrt{2\pi}} \int_{-\infty}^{\infty} e^{-i\omega t} y(\omega) \, dy,$$  
(16.2)

which has almost the same form as the original (forward) transform, apart from the sign of $i$ in the exponential.

This chapter is concerned with the discrete analog of Eqs. (16.1) and (16.2) in which the data $x_m$ is at a set of $N$ equally spaced “times”, and the Fourier transform $y_k$ is at a set of $N$ equally spaced “frequencies”. Note that in addition to the data being discrete, it only covers a finite range, whereas the original, continuous Fourier Transform extends to $\pm \infty$.

The discrete Fourier Transform is discussed in Sec. 16.2 in which the analogs of Eqs. (16.1) and (16.2) are Eqs. (16.3) and (16.4).

If $N$ is large the standard way of computing the transform, by directly evaluating Eq. (16.3) below for each of the $N$ values of the integer $k$, is very time consuming. In the appendices of this chapter we describe the fast Fourier transform (FFT) which is a much more efficient way to calculate a discrete Fourier transform. We don’t need the FFT for this course, but I include a description of it here in the appendices partly to stimulate students’ interest in it (since it is a gem of computer science), and partly because it bears a strong resemblance to Shor’s quantum Fourier transform (QFT), see Chapter 17, which is the heart of his factoring algorithm. We shall show this connection in the appendices of Chapter 17. In this chapter we deal exclusively with classical algorithms.

101
16.2 The Discrete Fourier Transform

If we have a set of \(N\) data points \(x_m\), the discrete Fourier transform (FT) is a set of \(N\) values \(y_k\) given by

\[
y_k = \frac{1}{\sqrt{N}} \sum_{m=0}^{N-1} \exp\left(\frac{2\pi i km}{N}\right) x_m ,
\]

(16.3)
evaluated for \(k = 0, 1, \cdots, N - 1\). The inverse Fourier transform has almost the same form; one just needs to take the complex conjugate of the exponential, i.e.

\[
x_m = \frac{1}{\sqrt{N}} \sum_{k=0}^{N-1} \exp\left(-\frac{2\pi i km}{N}\right) y_k .
\]

(16.4)

To see this we substitute Eq. (16.3) into Eq. (16.4) so

\[
x_m = \frac{1}{\sqrt{N}} \sum_{k=0}^{N-1} \exp\left(-\frac{2\pi i km}{N}\right) \frac{1}{\sqrt{N}} \sum_{l=0}^{N-1} \exp\left(\frac{2\pi i kl}{N}\right) x_l
\]

\[
= \frac{1}{N} \sum_{l=0}^{N-1} x_l \left[ \sum_{k=0}^{N-1} \exp\left(\frac{2\pi i k(l - m)}{N}\right) \right]
\]

\[
= \frac{1}{N} \sum_{l=0}^{N-1} x_l \left[ \frac{1 - \exp(2\pi i (l - m))}{1 - \exp(2\pi i (l - m)/N)} \right] ,
\]

(16.5)

where, in the last expression, we summed up the geometric series. The term in the rectangular brackets is 0 if \(l \neq m\). If \(l = m\) it is 0/0 so we either evaluate it as the limit \(l \to m\) or go back the start and put \(l = m\) from the beginning. In either method one finds that the term in rectangular brackets is equal to \(N\). Hence the RHS of Eq. (16.5) is \(x_m\), showing that the inverse transform in Eq. (16.4) does give back the original dataset \(x_m\) as claimed.

The discrete Fourier transform can be conveniently written as

\[
y_k = \frac{1}{\sqrt{N}} \sum_{m=0}^{N-1} \omega^{km} x_m ,
\]

(16.6)

where

\[
\omega = \exp\left(\frac{2\pi i}{N}\right).
\]

(16.7)

To determine the FT, each application of Eq. (16.6) requires \(N\) additions and \(N\) multiplications, for each of the \(N\) values of \(k\), so the operation count is \(O(N^2)\).

The Fast Fourier Transform (FFT) is able to evaluate this much more quickly, with an operation count of only \(N \log_2 N\), as least if \(N\) is a power of 2, i.e.

\[
N = 2^n .
\]

(16.8)

This turns out to be possible because \(\omega\) is a periodic function with period \(N\) and so \(\omega^{km}\) only takes \(N\) distinct values, even though \(km\) runs over \(O(N^2)\) values.

The FFT is discussed in the appendices which now follow. This material is not required for the rest of the course and can be omitted.
16.A. The Fast Fourier Transform; an example with \( N = 8 \)

We will understand the Fast Fourier Transform (FFT) by first working out in detail a simple example. The number of data points \( N \) must be a power of 2, see Eq. (16.8), and we will take \( n = 3 \), i.e. \( N = 8 \). Written out explicitly, the Fourier Transform for \( N = 8 \) data points is

\[
y_0 = \frac{1}{\sqrt{8}} \left( x_0 + x_1 + x_2 + x_3 + x_4 + x_5 + x_6 + x_7 \right), \quad (16.9a)
y_1 = \frac{1}{\sqrt{8}} \left( x_0 + \omega x_1 + \omega^2 x_2 + \omega^3 x_3 + \omega^4 x_4 + \omega^5 x_5 + \omega^6 x_6 + \omega^7 x_7 \right), \quad (16.9b)
y_2 = \frac{1}{\sqrt{8}} \left( x_0 + \omega^2 x_1 + \omega^4 x_2 + \omega^6 x_3 + x_4 + \omega^2 x_5 + \omega^4 x_6 + \omega^6 x_7 \right), \quad (16.9c)
y_3 = \frac{1}{\sqrt{8}} \left( x_0 + \omega^3 x_1 + \omega^6 x_2 + \omega x_3 + \omega^4 x_4 + \omega^7 x_5 + \omega^2 x_6 + \omega^5 x_7 \right), \quad (16.9d)
y_4 = \frac{1}{\sqrt{8}} \left( x_0 + \omega^4 x_1 + x_2 + \omega^4 x_3 + x_4 + \omega^4 x_5 + x_6 + \omega^4 x_7 \right), \quad (16.9e)
y_5 = \frac{1}{\sqrt{8}} \left( x_0 + \omega^5 x_1 + \omega^2 x_2 + \omega^7 x_3 + \omega^4 x_4 + \omega x_5 + \omega^6 x_6 + \omega^3 x_7 \right), \quad (16.9f)
y_6 = \frac{1}{\sqrt{8}} \left( x_0 + \omega^6 x_1 + \omega^4 x_2 + \omega^2 x_3 + x_4 + \omega^6 x_5 + \omega^4 x_6 + \omega^2 x_7 \right), \quad (16.9g)
y_7 = \frac{1}{\sqrt{8}} \left( x_0 + \omega^7 x_1 + \omega^6 x_2 + \omega^5 x_3 + \omega^4 x_4 + \omega^3 x_5 + \omega^2 x_6 + \omega x_7 \right), \quad (16.9h)
\]

where the \( x_j \) are the original data, the \( y_j \) are the Fourier transformed data,

\[
\omega = \exp(2\pi i/8) = \frac{1}{\sqrt{2}} (1 + i), \quad (16.10)
\]

and we note that

\[
\omega^8 = 1 = \omega^0, \quad (16.11)
\]

so we have reduced all the powers of \( \omega \) to be between 0 and 7 (\( = N - 1 \)). We also note that

\[
\omega^2 = i, \quad \omega^4 = -1. \quad (16.12)
\]

To evaluate Eqs. (16.9) efficiently the FFT proceeds recursively. We firstly define Fourier transforms of length 2:

\[
u_0 = \frac{1}{\sqrt{2}} (x_0 + x_4) = \frac{1}{\sqrt{2}} (x_0 + \omega^{4k} x_4) \quad (k = 0), \quad (16.13a)
u_1 = \frac{1}{\sqrt{2}} (x_1 + x_5) = \frac{1}{\sqrt{2}} (x_1 + \omega^{4k} x_5) \quad (k = 0), \quad (16.13b)
u_2 = \frac{1}{\sqrt{2}} (x_2 + x_6) = \frac{1}{\sqrt{2}} (x_2 + \omega^{4k} x_6) \quad (k = 0), \quad (16.13c)
u_3 = \frac{1}{\sqrt{2}} (x_3 + x_7) = \frac{1}{\sqrt{2}} (x_3 + \omega^{4k} x_7) \quad (k = 0), \quad (16.13d)
u_4 = \frac{1}{\sqrt{2}} (x_0 - x_4) = \frac{1}{\sqrt{2}} (x_0 + \omega^{4k} x_4) \quad (k = 1), \quad (16.13e)
u_5 = \frac{1}{\sqrt{2}} (x_1 - x_5) = \frac{1}{\sqrt{2}} (x_1 + \omega^{4k} x_5) \quad (k = 1), \quad (16.13f)
u_6 = \frac{1}{\sqrt{2}} (x_2 - x_6) = \frac{1}{\sqrt{2}} (x_2 + \omega^{4k} x_6) \quad (k = 1), \quad (16.13g)
u_7 = \frac{1}{\sqrt{2}} (x_3 - x_7) = \frac{1}{\sqrt{2}} (x_3 + \omega^{4k} x_7) \quad (k = 1). \quad (16.13h)
Pairs of quantities in Eqs. (16.13) are combined into Fourier Transforms of length 4:

\[ y_0 = \frac{1}{\sqrt{2}} (v_0 + v_1) = \frac{1}{\sqrt{2}} (v_0 + \omega^k v_1) \quad (k = 0), \]
\[ y_1 = \frac{1}{\sqrt{2}} (v_2 + \omega v_3) = \frac{1}{\sqrt{2}} (v_2 + \omega^k v_3) \quad (k = 1), \]
\[ y_2 = \frac{1}{\sqrt{2}} (v_4 + i\omega v_5) = \frac{1}{\sqrt{2}} (v_4 + \omega^k v_5) \quad (k = 2), \]
\[ y_3 = \frac{1}{\sqrt{2}} (v_6 + \omega^3 v_7) = \frac{1}{\sqrt{2}} (v_6 + \omega^k v_7) \quad (k = 3), \]
\[ y_4 = \frac{1}{\sqrt{2}} (v_0 - v_1) = \frac{1}{\sqrt{2}} (v_0 + \omega^k v_1) \quad (k = 4), \]
\[ y_5 = \frac{1}{\sqrt{2}} (v_2 - \omega v_3) = \frac{1}{\sqrt{2}} (v_2 + \omega^k v_3) \quad (k = 5), \]
\[ y_6 = \frac{1}{\sqrt{2}} (v_4 - i\omega v_5) = \frac{1}{\sqrt{2}} (v_4 + \omega^k v_5) \quad (k = 6), \]
\[ y_7 = \frac{1}{\sqrt{2}} (v_6 - \omega^3 v_7) = \frac{1}{\sqrt{2}} (v_6 + \omega^k v_7) \quad (k = 7), \]

and finally pairs of quantities in Eqs. (16.14) are combined to form the Fourier Transform in Eqs. (16.9):

\[ y_0 = \frac{1}{\sqrt{2}} (v_0 + u_2) = \frac{1}{\sqrt{2}} (u_0 + \omega^{2k} u_2) \quad (k = 0), \]
\[ y_1 = \frac{1}{\sqrt{2}} (v_1 + u_3) = \frac{1}{\sqrt{2}} (u_1 + \omega^{2k} u_3) \quad (k = 0), \]
\[ y_2 = \frac{1}{\sqrt{2}} (v_4 + i\omega u_6) = \frac{1}{\sqrt{2}} (u_4 + \omega^{2k} u_6) \quad (k = 1), \]
\[ y_3 = \frac{1}{\sqrt{2}} (v_5 + i\omega u_7) = \frac{1}{\sqrt{2}} (u_5 + \omega^{2k} u_7) \quad (k = 1), \]
\[ y_4 = \frac{1}{\sqrt{2}} (v_0 - u_2) = \frac{1}{\sqrt{2}} (u_0 + \omega^{2k} u_2) \quad (k = 2), \]
\[ y_5 = \frac{1}{\sqrt{2}} (v_1 - u_3) = \frac{1}{\sqrt{2}} (u_1 + \omega^{2k} u_3) \quad (k = 2), \]
\[ y_6 = \frac{1}{\sqrt{2}} (v_4 - i\omega u_6) = \frac{1}{\sqrt{2}} (u_4 + \omega^{2k} u_6) \quad (k = 3), \]
\[ y_7 = \frac{1}{\sqrt{2}} (v_5 - i\omega u_7) = \frac{1}{\sqrt{2}} (u_5 + \omega^{2k} u_7) \quad (k = 3), \]

Equations (16.13)–(16.15) are represented graphically by Fig. 16.1.

We see that the FFT, specified by Eqs. (16.13)–(16.15), requires \( 8 \times 3 = N \log_2 N \) for \( N = 8 \) additions and multiplications, whereas a direct evaluation of the FT according to Eq. (16.9) takes \( 8 \times 8 = N^2 \) additions and multiplications. For large \( N \), the speedup factor, \( N/\log_2 N \), in using the FFT rather than direct evaluation of the FT is considerable.

Let’s check that this works by evaluating \( y_1 \). We have

\[ y_1 = \frac{1}{\sqrt{2}} (v_2 + \omega v_3), \]
\[ = \frac{1}{2} \left( u_4 + i\omega u_6 + \omega (u_5 + i\omega u_7) \right) = \frac{1}{2} \left( u_4 + \omega^2 u_6 + \omega u_5 + \omega^3 u_7 \right), \]
\[ = \frac{1}{\sqrt{2}} \left( x_0 - x_4 + \omega^2 (x_2 - x_6) + \omega (x_1 - x_5) + \omega^3 (x_3 - u_7) \right), \]
\[ = \frac{1}{\sqrt{8}} \left( x_0 + \omega x_1 + \omega^2 x_2 + \omega^3 x_3 + \omega^4 x_4 + \omega^5 x_5 + \omega^6 x_6 + \omega^7 x_7 \right), \]

which agrees with Eq. (16.9b). We have used Eq. (16.15b) to get Eq. (16.16a), Eqs. (16.14c) and (16.14d) to get Eq. (16.16b), and Eqs. (16.13c), (16.13g), (16.13f) and (16.13h) to get Eq. (16.16c). Equation (16.16d) is the same as Eq. (16.16c) with powers of \( \omega \) written out explicitly using Eq. (16.12).

It is instructive to write the linear transformations in Eqs. (16.9), (16.13), (16.14) and (16.15) in matrix form. Equation (16.9) is written in matrix formulation as

\[ \tilde{y} = U \tilde{x}, \]
16.A. THE FAST FOURIER TRANSFORM; AN EXAMPLE WITH $N = 8$

The Fast Fourier Transform (FFT) is a fundamental algorithm used for efficiently computing the discrete Fourier transform (DFT) of a sequence. A graphical representation of the FFT for $N = 8$ is shown in Figure 16.1. The original data are the $x_j$ and the Fourier transformed data are the $y_j$. The dashed (red) lines have a factor of $-1$. The thick (green) circle transmits a factor of $\omega$ to the right, the dashed (blue) circles transmit a factor of $\omega^2 (= i)$ to the right, and the (brown) filled-in circle transmits a factor of $\omega^3$ to the right. In Sec. 16.C we will change to a notation applicable for general $n$, as follows: $y_j \equiv x_j^{(0)}, v_j \equiv x_j^{(1)}, u_j \equiv x_j^{(2)},$ and $x_j = x_j^{(3)}$. (Adapted from R. Vathsan Introduction to Quantum Physics and Information Processing.)

where

$$U = \frac{1}{\sqrt{8}} \begin{pmatrix} 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \\ 1 & \omega & \omega^2 & \omega^3 & \omega^4 & \omega^5 & \omega^6 & \omega^7 \\ 1 & \omega^2 & \omega^4 & \omega^6 & 1 & \omega^2 & \omega^4 & \omega^6 \\ 1 & \omega^3 & \omega^6 & \omega & \omega^4 & \omega^7 & \omega^2 & \omega^5 \\ 1 & \omega^4 & 1 & \omega^4 & 1 & \omega^4 & 1 & \omega^4 \\ 1 & \omega^5 & \omega^2 & \omega^7 & \omega^4 & \omega & \omega^6 & \omega^3 \\ 1 & \omega^6 & \omega^4 & \omega^2 & 1 & \omega^6 & \omega^4 & \omega^2 \\ 1 & \omega^7 & \omega^6 & \omega^5 & \omega^4 & \omega^3 & \omega^2 & \omega \end{pmatrix}.$$  \hspace{1cm} (16.18)

Equation (16.13) in matrix form is

$$\bar{u} = D\bar{x},$$  \hspace{1cm} (16.19)
where
\[
D = \frac{1}{\sqrt{2}} \begin{pmatrix}
1 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \\
0 & 1 & 0 & 0 & 0 & 1 & 0 & 0 \\
0 & 0 & 1 & 0 & 0 & 0 & 1 & 0 \\
0 & 0 & 0 & 1 & 0 & 0 & 0 & 1 \\
1 & 0 & 0 & 0 & \omega^4 & 0 & 0 & 0 \\
0 & 1 & 0 & 0 & 0 & \omega^4 & 0 & 0 \\
0 & 0 & 1 & 0 & 0 & 0 & \omega^4 & 0 \\
0 & 0 & 0 & 1 & 0 & 0 & 0 & \omega^4
\end{pmatrix}
\] (16.20)

Equation (16.14) in matrix form is
\[
\vec{v} = E\vec{u},
\] (16.21)
where
\[
E = \frac{1}{\sqrt{2}} \begin{pmatrix}
1 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\
0 & 1 & 0 & 1 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 1 & 0 & \omega^2 & 0 \\
0 & 0 & 0 & 0 & 1 & 0 & \omega^2 & 0 \\
1 & 0 & \omega^4 & 0 & 0 & 0 & 0 & 0 \\
0 & 1 & 0 & \omega^4 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 1 & 0 & \omega^6 & 0 & 0 \\
0 & 0 & 0 & 0 & 1 & 0 & \omega^6
\end{pmatrix}
\] (16.22)

Equation (16.15) in matrix form is
\[
\vec{y} = F\vec{v},
\] (16.23)
where
\[
F = \frac{1}{\sqrt{2}} \begin{pmatrix}
1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 1 & \omega & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 1 & \omega^2 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 1 & \omega^3 \\
1 & \omega^4 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 1 & \omega^5 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 1 & \omega^6 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 1 & \omega^7
\end{pmatrix}
\] (16.24)

Notice that \(D, E\) and \(F\), which describe the FFT, are very sparse, they have only two entries in each row and column, so they can be multiplied very efficiently, whereas the matrix \(U\), which describes the original Fourier transform, is dense. With some tedious matrix manipulations one can verify that
\[
U = FED,
\] (16.25)
as required. (I used Mathematica.)

16. B  Beyond \(N = 8\)

Now we discuss how we obtained Eqs. (16.13)–(16.15). For a general value \(n\), with \(N = 2^n\), the FT is defined by
\[
y_k = \frac{1}{\sqrt{N}} \sum_{m=0}^{N-1} \omega^{km} x_m, \quad (k = 0, 1, \ldots, N - 1)
\] (16.26)
with $\omega$ given by Eq. (16.7). We can break Eq. (16.26) into even and odd terms as follows:

$$y_k = \frac{1}{\sqrt{N}} \left[ \sum_{m=0}^{N/2-1} \omega^{2km} x_{2m} + \sum_{m=0}^{N/2-1} \omega^{k(2m+1)} x_{2m+1} \right],$$

$$= \frac{1}{\sqrt{2}} \left[ \sqrt{2} \sum_{m=0}^{N/2-1} (\omega^2)^{km} x_{2m} + \omega^k \sqrt{2} \sum_{m=0}^{N/2-1} (\omega^2)^{km} x_{2m+1} \right], \quad (k = 0,1,\ldots,N-1). \quad (16.27)$$

Noting that $\omega^2$ is the complex exponential factor analogous to Eq. (16.7) which figures in a Fourier Transform with $N/2$ points, we see that the first term in Eq. (16.27) is a FT for the $N/2$ even points and the second term is the FT for the $N/2$ odd points. We can write Eq. (16.27) as

$$y_k = \frac{1}{\sqrt{2}} \left[ v_{2k} + \omega^k v_{2k+1} \right], \quad (k = 0,1,\ldots,N-1), \quad (16.28)$$

where

$$v_{2k} = \sqrt{\frac{2}{N}} \sum_{m=0}^{N/2-1} (\omega^2)^{km} x_{2m},$$

$$v_{2k+1} = \sqrt{\frac{2}{N}} \sum_{m=0}^{N/2-1} (\omega^2)^{km} x_{2m+1}, \quad (k = 0,1,\ldots,N-1). \quad (16.29a)$$

Here $k$ runs over the range $0,1,\ldots,N-1$ so the indices on the $v_j$ in Eqs. (16.29) run from 0 to $2N-1$. However, since $\omega^N = 1$, see Eq. (16.7), it follows from the definition of the $v_j$ in Eq. (16.29) that $v_{j+N} = v_j$. Hence the index $j$, of the $v_j$ is to be evaluated modulo $N$. This applies in an obvious way to other quantities as well, such as the $u_j$, and, in Sec. 16.C, the lower index on the $x_j^{(r)}$.

For $N = 8$ please check that Eq. (16.28) corresponds to Eqs. (16.15) for $k = 0,1,2,\ldots,7$ and that, according to Eqs. (16.29), the expressions for the $v_k$ in terms of the original data $x_m$ are

$$v_0 = \frac{1}{2} \sum_{m=0}^{3} x_{2m}, \quad v_2 = \frac{1}{2} \sum_{m=0}^{3} (\omega^2)^{m} x_{2m}, \quad v_4 = \frac{1}{2} \sum_{m=0}^{3} (\omega^2)^{2m} x_{2m}, \quad v_6 = \frac{1}{2} \sum_{m=0}^{3} (\omega^2)^{3m} x_{2m},$$

$$v_1 = \frac{1}{2} \sum_{m=0}^{3} x_{2m+1}, \quad v_3 = \frac{1}{2} \sum_{m=0}^{3} (\omega^2)^{m} x_{2m+1}, \quad v_5 = \frac{1}{2} \sum_{m=0}^{3} (\omega^2)^{2m} x_{2m+1}, \quad v_7 = \frac{1}{2} \sum_{m=0}^{3} (\omega^2)^{3m} x_{2m+1}, \quad (16.30a)$$

so $v_0$, $v_2$, $v_4$ and $v_6$ are the FT of the 4 even points for $k = 0$, 1, 2 and 3 respectively, while $v_1$, $v_3$, $v_5$ and $v_7$ are the FT of the 4 odd points for $k = 0$, 1, 2 and 3 respectively.

We can again separate each of Eqs. (16.29) into even and odd terms by analogy with Eq. (16.27). We have

$$v_{2k} = \sqrt{\frac{2}{N}} \left[ \sum_{m=0}^{N/4-1} (\omega^4)^{km} x_{4m} + (\omega^2)^k \sum_{m=0}^{N/4-1} (\omega^4)^{km} x_{4m+2} \right],$$

$$v_{2k+1} = \sqrt{\frac{2}{N}} \left[ \sum_{m=0}^{N/4-1} (\omega^4)^{km} x_{4m+1} + (\omega^2)^k \sum_{m=0}^{N/4-1} (\omega^4)^{km} x_{4m+3} \right]. \quad (16.31a)$$
We can write these equations as
\begin{equation}
    v_{2k} = \frac{1}{\sqrt{2}} \left[ u_{4k} + (\omega^2)^k u_{4k+2} \right], \quad (k = 0, 1, \cdots, N/2 - 1),
    \tag{16.32a}
\end{equation}
\begin{equation}
    v_{2k+1} = \frac{1}{\sqrt{2}} \left[ u_{4k+1} + (\omega^2)^k u_{4k+3} \right], \quad (k = 0, 1, \cdots, N/2 - 1),
    \tag{16.32b}
\end{equation}

where
\begin{equation}
    u_{4k} = \sqrt{\frac{4}{N}} \sum_{m=0}^{N/4-1} (\omega^4)^{km} x_{4m}, \quad u_{4k+1} = \sqrt{\frac{4}{N}} \sum_{m=0}^{N/4-1} (\omega^4)^{km} x_{4m+1},
    \tag{16.33a}
\end{equation}
\begin{equation}
    u_{4k+2} = \sqrt{\frac{4}{N}} \sum_{m=0}^{N/4-1} (\omega^4)^{km} x_{4m+2}, \quad u_{4k+3} = \sqrt{\frac{4}{N}} \sum_{m=0}^{N/4-1} (\omega^4)^{km} x_{4m+3}.
    \tag{16.33b}
\end{equation}

Note that the two equations in Eqs. (16.32) can be combined as
\begin{equation}
    v_{2k+p} = \frac{1}{\sqrt{2}} \left[ u_{4k+p} + (\omega^2)^k u_{4k+p+2} \right], \quad (p = 0, 1), (k = 0, 1, \cdots, N/2 - 1).
    \tag{16.34}
\end{equation}

Again, the index \( j \) on the \( u_j \) is to be evaluated modulo \( N \).

For \( N = 8 \) please check that Eq. (16.34) corresponds to our Eqs. (16.14) for \( p = 0, 1 \), and \( k = 0, 1, 2 \) and 3, and that, according to Eqs. (16.33), the explicit expressions for the \( u_j \) are
\begin{equation}
    u_0 = \frac{1}{\sqrt{2}} \sum_{m=0}^{1} x_{4m} = \frac{1}{\sqrt{2}} (x_0 + x_4), \quad u_1 = \frac{1}{\sqrt{2}} \sum_{m=0}^{1} x_{4m+1} = \frac{1}{\sqrt{2}} (x_1 + x_5),
    \tag{16.35a}
\end{equation}
\begin{equation}
    u_2 = \frac{1}{\sqrt{2}} \sum_{m=0}^{1} x_{4m+2} = \frac{1}{\sqrt{2}} (x_2 + x_6), \quad u_3 = \frac{1}{\sqrt{2}} \sum_{m=0}^{1} x_{4m+3} = \frac{1}{\sqrt{2}} (x_3 + x_7),
    \tag{16.35b}
\end{equation}
\begin{equation}
    u_4 = \frac{1}{\sqrt{2}} \sum_{m=0}^{1} (\omega^4)^m x_{4m} = \frac{1}{\sqrt{2}} (x_0 - x_4), \quad u_5 = \frac{1}{\sqrt{2}} \sum_{m=0}^{1} (\omega^4)^m x_{4m+1} = \frac{1}{\sqrt{2}} (x_1 - x_5),
    \tag{16.35c}
\end{equation}
\begin{equation}
    u_6 = \frac{1}{\sqrt{2}} \sum_{m=0}^{1} (\omega^4)^m x_{4m+2} = \frac{1}{\sqrt{2}} (x_2 - x_6), \quad u_7 = \frac{1}{\sqrt{2}} \sum_{m=0}^{1} (\omega^4)^m x_{4m+3} = \frac{1}{\sqrt{2}} (x_3 - x_7).
    \tag{16.35d}
\end{equation}

Equations (16.35) agree with the expressions in Eq. (16.13). They can be written as a single equation as
\begin{equation}
    u_{4k+p} = \frac{1}{\sqrt{2}} [x_p + (-1)^k x_{p+4}], \quad (p = 0, 1, 2, 3), (k = 0, 1).
    \tag{16.36}
\end{equation}

Thus we have seen that the FFT for \( N = 8 (= 2^3) \) with \( n = 3 \), which is written out explicitly in Eqs. (16.13)–(16.15), corresponds to firstly doing the Fourier transforms of length 2 in Eq. (16.36), followed by two applications of the iterative procedure, the first shown in Eq. (16.34) and the second shown in Eq. (16.28).

16.C  The General Case

So far we have unsystematically labeled the results at each stage of iteration by a different symbol, \( x \to u \to v \to y \), see Fig. (16.1) When writing a code applicable for \( N = 2^n \) data points for arbitrary
n, one would use a common symbol but add a second index, so

\[ x_j \equiv x_j^{(n)} , \quad (16.37a) \]

\[ u_j \equiv x_j^{(2)} , \quad (16.37b) \]

\[ v_j \equiv x_j^{(1)} , \quad (16.37c) \]

\[ y_j \equiv x_j^{(0)} . \quad (16.37d) \]

Note that since \( \omega = \exp(2\pi i/2^n) \) we have

\[ \omega^{2^n} = \exp(2\pi i) = 1, \quad \omega^{2^{n-1}} = \exp(\pi i) = -1. \quad (16.38) \]

The \( \ell \)-th iteration, analogous to Eqs. (16.34), (16.28) and (16.36) is

\[ x^{(\ell-1)}_{2^\ell-1 k+p} = \frac{1}{\sqrt{2}} \left[ x^{(\ell)}_{2^\ell k+p} + (\omega^{2^\ell-1})^k x^{(\ell)}_{2^\ell k+p+2^\ell-1} \right], \quad (16.39) \]

with

\[ p = 0, 1, \cdots, 2^{\ell-1} - 1, \quad k = 0, 1, \cdots, 2^{n-\ell+1} - 1. \quad (16.40) \]

Sorry that the notation is messy but I can’t see how to improve it; one just has to keep track of the indices and the powers of \( \omega \). Recall that the lower index \( j \) on the \( x_j^{(\ell)} \) is to be evaluated modulo \( 2^n \).

Let’s see how this works.

- We start with \( \ell = n \), for which \( x_j^{(\ell)} \equiv x_j \), the original data points.
  Equation (16.39) is then

\[ x^{(n-1)}_{2^n-1 k+p} = \frac{1}{\sqrt{2}} \left[ x_p + (-1)^k x_{p+2^n-1} \right], \quad (p = 0, 1, \cdots, 2^{n-1} - 1, \quad k = 0, 1). \quad (16.41) \]

For \( n = 3 \) (\( N = 8 \)) this corresponds to Eq. (16.36) with \( x_j^{(n-1)} \equiv u_j \).

- We then iterate Eq. (16.39) for \( \ell = n-1, n-2, \cdots, 2, 1 \).
  At the next to the last iteration, \( \ell = 2 \), we have

\[ x^{(1)}_{2k+p} = \frac{1}{\sqrt{2}} \left[ x^{(2)}_{4k+p} + (\omega^2)^k x^{(2)}_{4k+p+2} \right], \quad (p = 0, 1), \quad (k = 0, 1, \cdots, 2^{n-1} - 1), \quad (16.42) \]

which corresponds to Eq. (16.34) with \( x_j^{(1)} \equiv v_j, x_j^{(2)} \equiv u_j \). At the last iteration, \( \ell = 1 \), we obtain

\[ y_k = \frac{1}{\sqrt{2}} \left[ x^{(1)}_{2k} + \omega^k x^{(1)}_{2k+1} \right], \quad (k = 0, 1, 2, \cdots, 2^n - 1), \quad (16.43) \]

which is Eq. (16.28). (Recall that \( x_j^{(0)} \equiv y_j \), the Fourier transformed data, and \( x_j^{(1)} \equiv v_j \).)

Note that the iterations are evaluated in reverse, starting with \( \ell = n \) and working down to \( \ell = 1 \).
Chapter 17

The Quantum Fourier Transform (QFT) and a Comparison with the FFT

17.1 Introduction

This chapter introduces the quantum Fourier transform (QFT), which is at the heart of Shor’s algorithm for period finding, and hence for factoring. Shor’s algorithm will be discussed in Chapter 18. The appendices make a detailed comparison with the (classical) Fast Fourier Transform (FFT). The FFT is not part of the course so if you are not interested in this comparison you can ignore the appendices.

17.2 Quantum Fourier Transform (QFT) for $N = 4$

In this section we describe a simple example, that of $N = 4$ states, i.e. $n = 2$ qubits.

Firstly, a reminder about notation. A state of $n$ qubits is labeled either by a single $n$-bit integer, $x$ say, or by the values of the individual qubits $x_i$, which form the binary representation of $x$. As an example, the four computational basis states for two qubits are

\[
|0\rangle_2 \equiv |00\rangle,
|1\rangle_2 \equiv |01\rangle,
|2\rangle_2 \equiv |10\rangle,
|3\rangle_2 \equiv |11\rangle,
\]

(17.1)

To prevent any ambiguity, we will, when necessary, indicate the number of qubits in a state by a subscript, e.g. $|x\rangle_n$ has $n$ qubits so $x$ will be an $n$-bit integer, whose bits indicate the state of the individual qubits.

We now describe the Quantum Fourier Transform (QFT) for $n = 2$ qubits.

In the Quantum Fourier Transform (QFT) one starts from an initial state $|x\rangle_2 \equiv |x_1x_0\rangle$ in the computational basis and generates the following superposition

\[
|\psi_x\rangle_2 = \frac{1}{2} \sum_{y=0}^{3} \exp \left[ 2\pi i xy/2^2 \right] |y\rangle_2,
\]

(17.2)

where $|y\rangle_2 \equiv |y_1y_0\rangle$. Note this has the same structure as a standard Fourier Transform, but here we are performing a change of basis of a quantum system, rather than transforming a set of data. The $|\psi_x\rangle_2$ form a basis just as the $|x\rangle_2$ form a basis because one can show that they are orthonormal, i.e.

\[
_2\langle x_1 | \psi_{x_2} \rangle_2 = \delta_{x_1,x_2}.
\]

(17.3)
Noting that $y = y_0 + 2y_1$ and $x = x_0 + 2x_1$ we can simplify the argument of the exponential:

$$\frac{2\pi i xy}{2^2} = \frac{2\pi i(x_0 + 2x_1)(y_0 + 2y_1)}{2^2} = 2\pi i \left\{ y_0 \left( \frac{x_0}{4} + \frac{x_1}{2} \right) + y_1 \left( \frac{x_0}{2} + x_1 \right) \right\}. \quad (17.4)$$

Now $\exp(2\pi iy_1x_1) = 1$ so the factor $y_1x_1$ above can be neglected. Hence Eq. (17.2) becomes

$$|\psi_x\rangle_2 = \left( \frac{1}{\sqrt{2}} \sum_{y_0=0}^1 \exp \left[ 2\pi iy_0 \left( \frac{x_0}{4} + \frac{x_1}{2} \right) \right] \right) \left( \frac{1}{\sqrt{2}} \sum_{y_1=0}^1 \exp \left[ 2\pi iy_1 \frac{x_0/2}{2} \right] \right) |y_1y_0\rangle. \quad (17.5)$$

Next we will explain how to perform the operations in Eq. (17.5) using quantum gates.

Consider the second factor on the RHS of Eq. (17.5), which involves a sum over $y_1$. The exponential is 1 for $y_1 = 0$ and is $-1$ for $y_1 = 1$ provided $x_0 = 1$. This functionality is provided by a Hadamard gate $H$,

$$H = \frac{1}{\sqrt{2}} \begin{pmatrix} 1 & 1 \\ 1 & -1 \end{pmatrix} \quad (17.6)$$

so

$$H|0\rangle = \frac{1}{\sqrt{2}}(|0\rangle + |1\rangle),$$
$$H|1\rangle = \frac{1}{\sqrt{2}}(|0\rangle - |1\rangle). \quad (17.7)$$

Recall that in terms of components

$$|0\rangle \equiv \begin{pmatrix} 1 \\ 0 \end{pmatrix}, \quad |1\rangle \equiv \begin{pmatrix} 0 \\ 1 \end{pmatrix}. \quad (17.8)$$

More compactly, we can write

$$H|x_0\rangle = \frac{1}{\sqrt{2}} (|0\rangle + (-1)^{x_0}|1\rangle) = \frac{1}{\sqrt{2}} \sum_{y_1=0}^1 (-1)^{x_0y_1} |y_1\rangle = \frac{1}{\sqrt{2}} \sum_{y_1=0}^1 \exp[2\pi iy_1x_0/2]|y_1\rangle, \quad (17.9)$$

where we denote by $y_1$ the dummy variable to be summed over (coming from the effects of the Hadamard) in order to correspond with the second factor on the RHS of Eq. (17.5). We therefore see that the second factor on the RHS of Eq. (17.5) is generated by the Hadamard gate shown in Fig. 17.1.

$$|x_0\rangle \xrightarrow{H} |y_1\rangle \xrightarrow{H} |\psi_0\rangle$$

Figure 17.1: The output from the Hadamard gate is $|\psi_0\rangle = \frac{1}{\sqrt{2}}(|y_1=0\rangle + (-1)^{x_0}|y_1=1\rangle)$. This can be expressed as $\frac{1}{\sqrt{2}} \sum_{y_1=0}^1 (-1)^{x_0y_1} |y_1\rangle = \frac{1}{\sqrt{2}} \sum_{y_1=0}^1 \exp[2\pi iy_1x_0/2]|y_1\rangle$. Recall that $x_0$ takes a fixed value 0 or 1.

What about the first factor on the RHS of Eq. (17.5) which involves $y_0$? There are two pieces in the exponential. The one involving $2\pi iy_0x_1/2$ can be dealt with by a Hadamard, similar to Fig. 17.1 but with the left hand qubit being $x_1$ and the right hand qubit being labeled by $y_0$. However, the piece
17.2. QUANTUM FOURIER TRANSFORM (QFT) FOR N = 4

involving $2\pi iy_0x_0/4$ is different. It induces a phase shift of $e^{i\pi/2}$ for $y_0 = 1$ provided that $x_0$ is also 1. This requires a controlled phase gate. We define a phase gate $R_d$ by

$$R_d = \begin{pmatrix} 1 & 0 \\ 0 & e^{i\pi/2^d} \end{pmatrix}. \quad (17.10)$$

Acting on $|0\rangle$, $R_d$ makes no change, while acting on $|1\rangle$ $R_d$ changes the phase by $\pi/2^d$. Note that $R_0$ is just the Z gate. Here we need $R_1$.

The exponential in the first term on the RHS of Eq. (17.5) can be generated by a Hadamard followed by a controlled $R_1$ gate as shown for the top qubit in Fig. 17.2. Note that the $R_1$ phase gate on the upper qubit is controlled by the lower qubit. Including the Hadamard on the lower qubit, Fig. 17.2 generates both factors on the RHS of Eq. (17.5).

Figure 17.2: The initial state on the left is the single quantum state $|x\rangle_2 \equiv |x_1x_0\rangle$ in the computational basis. The final state on the right is the superposition $|\psi'\rangle_2 = (1/2) \sum_y e^{2\pi i y_0x_1/2} e^{2\pi i y_0x_0/4} |y_0y_1\rangle$, which is almost $|\psi_x\rangle_2$, the QFT of $|x\rangle_2 \equiv |x_1x_0\rangle$ given in Eq. (17.5), except that the order of the bits in the final state is the reverse of what it should be according to Eq. (17.5). This can be corrected by a swap gate as shown in Fig. 17.3. Note the controlled-$R_1$ phase gate. The general phase gate $R_d$ is defined in Eq. (17.10).

To make sure we understand what is happening in the circuit in Fig. 17.2 we now write down the state at each of the steps shown in the figure. After the first Hadamard the state is

$$|\phi_1\rangle_2 = \frac{1}{\sqrt{2}} \sum_{y_0=0}^1 e^{2\pi i y_0 x_1/2} |y_0x_0\rangle, \quad (17.11a)$$

where we denote by $y_0$ the dummy variable that is summed over in order to agree with Eq. (17.5). After the controlled-$R_1$ gate we have

$$|\phi_2\rangle_2 = \frac{1}{\sqrt{2}} \sum_{y_0=0}^1 e^{2\pi i y_0 x_1/2} e^{2\pi i y_0 x_0/4} |y_0x_0\rangle. \quad (17.11b)$$

The final state after the Hadamard on the lower qubit is therefore

$$|\psi'\rangle_2 = \left( \frac{1}{\sqrt{2}} \sum_{y_0=0}^1 e^{2\pi i y_0 x_1/2} e^{2\pi i y_0 x_0/4} \right) \left( \frac{1}{\sqrt{2}} \sum_{y_1=0}^1 e^{2\pi i y_1 x_0/2} \right) |y_0y_1\rangle. \quad (17.11c)$$

---

1This is the definition of $R_d$ given in Vathsan’s book, and I find it convenient. Some other authors adopt a slightly different definition with $e^{2\pi i/2^d}$ instead of $e^{\pi i/2^d}$. 
\[ |\psi_x\rangle_3 = \frac{1}{\sqrt{8}} \sum_{y=0}^{7} \exp[2\pi i xy/2^3] |y\rangle_3 \]

\[ = \left( \frac{1}{\sqrt{2}} \sum_{y_0=0}^{1} \exp\left[2\pi i y_0 \left( \frac{x_0}{8} + \frac{x_1}{4} + \frac{x_2}{2} \right) \right] \right) \left( \frac{1}{\sqrt{2}} \sum_{y_1=0}^{1} \exp\left[2\pi i y_1 \left( \frac{x_0}{4} + \frac{x_1}{2} \right) \right] \right) \times \left( \frac{1}{\sqrt{2}} \sum_{y_2=0}^{1} \exp\left[2\pi i y_2 \frac{x_0}{2} \right] \right) |y_2 y_1 y_0\rangle. \]

Following along the lines in the previous section, the circuit diagram which will perform this is shown in Fig. 17.4. To make sure we understand this circuit we will write down the state at each stage indicated on the figure. (Although these expressions look rather complicated is useful to make the
17.3. QFT WITH THREE OR MORE QUBITS

Figure 17.4: Circuit diagram for performing the QFT with $n = 3$ qubits. It generates the transformation shown in Eq. (17.13). The phase gates, $R_d$, are defined in Eq. (17.10). The dashed line with crosses at the ends indicates a swap gate between qubits 0 and 2. This serves to reverse the order of the qubits.

$|\phi_1\rangle_3 = \left( \frac{1}{\sqrt{2}} \sum_{y_0=0}^{1} \exp \left[ 2\pi i y_0 \left( \frac{x_2}{2} \right) \right] \right) |y_0 x_1 x_2\rangle$,

(17.14a)

$|\phi_2\rangle_3 = \left( \frac{1}{\sqrt{2}} \sum_{y_0=0}^{1} \exp \left[ 2\pi i y_0 \left( \frac{x_0}{8} + \frac{x_1}{4} + \frac{x_2}{2} \right) \right] \right) |y_1 x_1 x_2\rangle$,

(17.14b)

$|\phi_3\rangle_3 = \left( \frac{1}{\sqrt{2}} \sum_{y_0=0}^{1} \exp \left[ 2\pi i y_0 \left( \frac{x_0}{8} + \frac{x_1}{4} + \frac{x_2}{2} \right) \right] \right) \left( \frac{1}{\sqrt{2}} \sum_{y_1=0}^{1} \exp \left[ 2\pi i y_1 \left( \frac{x_1}{2} \right) \right] \right) |y_0 y_1 x_2\rangle$,

(17.14c)

$|\phi_4\rangle_3 = \left( \frac{1}{\sqrt{2}} \sum_{y_0=0}^{1} \exp \left[ 2\pi i y_0 \left( \frac{x_0}{8} + \frac{x_1}{4} + \frac{x_2}{2} \right) \right] \right) \left( \frac{1}{\sqrt{2}} \sum_{y_1=0}^{1} \exp \left[ 2\pi i y_1 \left( \frac{x_0}{4} + \frac{x_1}{2} \right) \right] \right) \left( \frac{1}{\sqrt{2}} \sum_{y_2=0}^{1} \exp \left[ 2\pi i y_2 \left( \frac{x_0}{2} \right) \right] \right) |y_0 y_1 y_2\rangle$,

(17.14d)

$|\psi'_x\rangle_3 = \left( \frac{1}{\sqrt{2}} \sum_{y_0=0}^{1} \exp \left[ 2\pi i y_0 \left( \frac{x_0}{8} + \frac{x_1}{4} + \frac{x_2}{2} \right) \right] \right) \left( \frac{1}{\sqrt{2}} \sum_{y_1=0}^{1} \exp \left[ 2\pi i y_1 \left( \frac{x_0}{4} + \frac{x_1}{2} \right) \right] \right) \times \left( \frac{1}{\sqrt{2}} \sum_{y_2=0}^{1} \exp \left[ 2\pi i y_2 \left( \frac{x_0}{2} \right) \right] \right) |y_0 y_1 y_2\rangle$.

(17.14e)

$|\psi'_x\rangle$ is almost the desired QFT in Eq. (17.13), except that the order of the qubits on in the final state on the right has been reversed. This can be compensated for by adding a swap gate between qubits 1 and 3. Hence $|\psi_x\rangle$ in the figure is the desired QFT for 3 qubits given in Eq. (17.13).

Intuitively, the reason that for the reverse order of the qubits in the final state before the swaps, is the following. The Hadamards produce the factors in the exponential involving $2\pi i/2$. From Eq. (17.13), and generalizing Eq. (17.4) to arbitrary $n$, one sees that these are $(2\pi i/2) \sum_{j=0}^{n-1} x_j y_{n-j}$. 

...
CHAPTER 17. THE QUANTUM FOURIER TRANSFORM (QFT) AND A COMPARISON WITH THE FFT

Figure 17.5: Circuit diagram for performing the QFT with an arbitrary number of qubits. For clarity the final swaps are not shown, so the input states on the left, $x_i$, and the output states on the right, $y_i$, are in opposite order. Note that the controlled phase gate between qubits $x_i$ and $x_j$ is $R_{|i-j|}$, which makes the structure fairly simple.

Now $x_j$ is the label of the $j$-th qubit in the initial state, and $y_{n-j}$ is the dummy label (summed over) for the same qubit after the Hadamard and in the final state. Because it is $y_{n-j}$ which goes with $x_j$, rather than $y_j$, the qubits in the final state are in reverse order.

Comparing with the case for two qubits shown in Fig. 17.3 and that for three qubits in Fig. 17.4, the generalization to an arbitrary number of qubits, where

$$|\psi_x\rangle_n = \frac{1}{\sqrt{2^n}} \sum_{y=0}^{2^n-1} \exp[2\pi i xy/2^n]|y\rangle_n$$ (17.15)

can be deduced and is shown in Fig. 17.5. Note that the controlled phase gate between qubits $x_i$ and $x_j$ is $R_{|i-j|}$, which makes the structure fairly simple.

For an $n$-qubit QFT one needs $n$ Hadamard gates. The number of controlled phase gates is $1 + 2 + \cdots + n - 1 = n(n - 1)/2$. Also $[n/2]$ swaps are required, where $[k]$ denotes the largest integer less than or equal to $k$. The circuit therefore provides an algorithm for performing the QFT in $O(n^2)$ steps. By contrast the FFT requires $O(n2^n)$ steps which is exponentially greater.

However, we cannot obtain the $2^n$ Fourier amplitudes from the QFT since a measurement will just give one of the basis states with a probability proportional to the square of the absolute value of its Fourier amplitude. The QFT can, however, give useful information if the input state is a linear combination $\sum_x a_x |x\rangle$ where the $a_x$ are periodic in $x$ with some period $r$. As we shall see in Chapter 18 the Fourier amplitudes are then strongly peaked at multiples of $2^n/r$, so there is a high probability of getting a value for $y$ at, or close to, a multiple of $2^n/r$, from which it turns out that one can deduce $r$ with high probability. Hence the QFT is useful for period finding.

As we saw in Chapter 15, period finding can be used to factor integers. If one could factor large integers, one would be able to decode messages sent down the internet which have been encoded with the standard RSA encryption method. We discussed RSA encryption in Chapter 14.

In the appendices we describe the connection between the QFT and FFT. This material is not necessary for the rest of the course and can be skipped.
Appendices

17.A Fast Fourier Transform (FFT) for \( N = 4 \)

For comparison with the QFT we write out the Fast Fourier Transform (FFT) for \( N = 4 \). The Fourier transform for this case is

\[
y_0 = \frac{1}{2} \left( x_0 + x_1 + x_2 + x_3 \right), \tag{17.16a}
\]

\[
y_1 = \frac{1}{2} \left( x_0 + ix_1 + i^2 x_2 + i^3 x_3 \right), \tag{17.16b}
\]

\[
y_2 = \frac{1}{2} \left( x_0 + i^2 x_1 + x_2 + i^2 x_3 \right), \tag{17.16c}
\]

\[
y_3 = \frac{1}{2} \left( x_0 + i^3 x_1 + i^2 x_2 + ix_3 \right), \tag{17.16d}
\]

where the \( x_j \) are the original data, the \( y_j \) are the Fourier transformed data, and we have used that

\[
\exp(2\pi i/4) = i. \tag{17.17}
\]

To evaluate Eqs. (17.16) efficiently, the FFT proceeds recursively. We firstly define Fourier transforms of length 2:

\[
u_0 = \frac{1}{\sqrt{2}} (x_0 + x_2) = \frac{1}{\sqrt{2}} (x_0 + i^{2k} x_2) \quad (k = 0), \tag{17.18a}
\]

\[
u_1 = \frac{1}{\sqrt{2}} (x_1 + x_3) = \frac{1}{\sqrt{2}} (x_1 + i^{2k} x_3) \quad (k = 0), \tag{17.18b}
\]

\[
u_2 = \frac{1}{\sqrt{2}} (x_0 - x_2) = \frac{1}{\sqrt{2}} (x_0 + i^{2k} x_2) \quad (k = 1), \tag{17.18c}
\]

\[
u_3 = \frac{1}{\sqrt{2}} (x_1 - x_3) = \frac{1}{\sqrt{2}} (x_1 + i^{2k} x_3) \quad (k = 1), \tag{17.18d}
\]

Pairs of quantities in Eqs. (17.18) are combined to form the Fourier Transform in Eqs. (17.16):

\[
y_0 = \frac{1}{\sqrt{2}} (u_0 + u_1) = \frac{1}{\sqrt{2}} (u_0 + i^k u_1) \quad (k = 0), \tag{17.19a}
\]

\[
y_1 = \frac{1}{\sqrt{2}} (u_2 + i u_3) = \frac{1}{\sqrt{2}} (u_2 + i^k u_3) \quad (k = 1), \tag{17.19b}
\]

\[
y_2 = \frac{1}{\sqrt{2}} (u_0 - u_1) = \frac{1}{\sqrt{2}} (u_0 + i^k u_1) \quad (k = 2), \tag{17.19c}
\]

\[
y_3 = \frac{1}{\sqrt{2}} (u_2 - i u_3) = \frac{1}{\sqrt{2}} (u_2 + i^k u_3) \quad (k = 3), \tag{17.19d}
\]

Let’s check that this works by evaluating \( y_1 \). We have

\[
y_1 = \frac{1}{\sqrt{2}} (u_2 + i u_3), \tag{17.20a}
\]

\[
= \frac{1}{2} \left( x_0 - x_2 + i (x_1 - x_3) \right) = \frac{1}{2} \left( u_0 + ix_1 + i^2 x_2 + i^3 x_3 \right), \tag{17.20b}
\]

which agrees with Eq. (17.16b).

It is instructive to write the linear transformations in Eqs. (17.16), (17.18), and (17.19) in matrix form. Equation (17.16) is written in matrix formulation as

\[
\vec{y} = U \vec{x}, \tag{17.21}
\]

where

\[
U = \frac{1}{2} \begin{pmatrix}
1 & 1 & 1 & 1 \\
1 & i & i^2 & i^3 \\
i^2 & 1 & i & i^2 \\
i^3 & i^2 & i & i
\end{pmatrix}. \tag{17.22}
\]
Equation (17.18) in matrix form is
\[ \vec{u} = U_1 \vec{x}, \]  
(17.23)
where
\[ U_1 = \frac{1}{\sqrt{2}} \begin{pmatrix} 1 & 0 & 1 & 0 \\ 0 & 1 & 0 & 1 \\ 1 & 0 & i^2 & 0 \\ 0 & 1 & 0 & i^2 \end{pmatrix}. \]  
(17.24)

Equation (17.19) in matrix form is
\[ \vec{y} = U_2 \vec{u}, \]  
(17.25)
where
\[ U_2 = \frac{1}{\sqrt{2}} \begin{pmatrix} 1 & 1 & 0 & 0 \\ 0 & 0 & 1 & i \\ 1 & i^2 & 0 & 0 \\ 0 & 0 & 1 & i^3 \end{pmatrix}. \]  
(17.26)

With some matrix manipulations one can verify that
\[ U = U_2 U_1, \]  
(17.27)
as required. (I used Mathematica.)

17.B Comparison between FFT and the QFT for \( N = 4 \)

The QFT is generated by the unitary matrix \( U \) in Eq. (17.22). The classical FFT is written as the product of two sparse matrices, \( U = U_2 U_1 \), see Eq. (17.27), where \( U_1 \) is given in Eq. (17.24) and \( U_2 \) is given in Eq. (17.26). We will now see that there is a close connection between the FFT and the QFT, in particular, that the transformations \( U_1 \) and \( U_2 \) correspond to different parts of the diagram in Fig. 17.3.

The swap gate interchanges states \( |1\rangle \) and \( |2\rangle \), see Eq. 17.1, so it has the matrix representation
\[ S = \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix}. \]  
(17.28)

The Hadamard gate acting on the lower qubit of Fig. 17.3 was shown in Eq. (11.10). Including now also the (unchanged) upper qubit, the matrix representation of the transformation induced by this gate is
\[ H_l = \frac{1}{\sqrt{2}} \begin{pmatrix} 1 & 1 & 0 & 0 \\ 1 & -1 & 0 & 0 \\ 0 & 0 & 1 & 1 \\ 0 & 0 & 1 & -1 \end{pmatrix}. \]  
(17.29)

The Hadamard on the upper qubit has a similar representation, except that the two qubits have been interchanged, i.e.
\[ H_u = \frac{1}{\sqrt{2}} \begin{pmatrix} 1 & 0 & 1 & 0 \\ 0 & 1 & 0 & 1 \\ 1 & 0 & -1 & 0 \\ 0 & 1 & 0 & -1 \end{pmatrix}. \]  
(17.30)
The controlled $R_1$ phase gate gives a multiplicative factor of $i$ if $y_0$ and $x_0$ are both 1, i.e. state $|3\rangle$. Hence

$$R_1 = \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & i \end{pmatrix}.$$  \hfill (17.31)

The total effect of the quantum circuit in Fig. 17.3 reading from left to right on the circuit, is given by the matrix product $SH_l R_1 H_u$. Note that one reads from right to left in a product of operators because the operators act on the right. It can be confusing that the direction of time in the circuit diagram is opposite to that in an expression of operators. Multiplying these out these matrices using Mathematica one gets the expected result,

$$SH_l R_1 H_u = U,$$  \hfill (17.32)

where $U$ is the Fourier transform, shown in Eq. (17.22). Recall that $S$ is the swap, $H_l$ is the Hadamard on the lower qubit, $R_1$ is the controlled phase gate, and $H_u$ is the Hadamard on the upper qubit. Hence the gates in the quantum circuit in Fig. 17.3 do indeed affect a Fourier transform for 2 qubits.

In the FFT we decomposed $U$ into a product of two sparse matrices, $U = U_2 U_1$, see Eq. (17.27). We can also make a connection between the individual matrices $U_2$ and $U_1$ of the FFT and the individual matrices $S, H_l, H_u$ and $R_1$ of the QFT. One finds

$$U_1 = H_u,$$  \hfill (17.33a)

$$U_2 = SH_l R_1.$$  \hfill (17.33b)

The first is obtained by inspection and the second I checked with Mathematica. Hence the first operation $U_1$ in the FFT for $N = 4$ corresponds, in the QFT, to the Hadamard on the upper qubit in Fig. 17.3 while the second operation $U_2$ in the FFT corresponds to the remaining operations in the QFT: the controlled phase gate on the upper qubit, the Hadamard on the lower qubit, and the swap. This breakup is shown in Fig. 17.6.

To conclude this section, we have seen that there is close connection between the breakup used in the FFT and that used in the QFT. This should not be a surprise. In the FFT we iteratively divide the FT into two FTs of half the length, while in the QFT we have a binary representation of the states and treat each bit in turn, so clearly there is a connection. For $N = 4$, this connection is expressed in Eqs. (17.33).
17.C Comparison of FFT and QFT for \( N = 8 \)

In this appendix we show how the breakup of the FFT for \( N = 8 \) is related to the circuit for the QFT. Our final result will be Fig. 17.7, which is the analog of Fig. 17.6 for \( N = 4 \).

As shown in Chapter 16, the FFT for \( N = 8 \) can be written as

\[
U^{(8)} = U_1^{(8)} U_2^{(8)} U_3^{(8)}
\]  (17.34)

where

\[
U^{(8)} = \frac{1}{\sqrt{8}} \begin{pmatrix}
1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \\
1 & \omega & \omega^2 & \omega^3 & \omega^4 & \omega^5 & \omega^6 & \omega^7 \\
1 & \omega^2 & \omega^4 & \omega^6 & 1 & \omega^2 & \omega^4 & \omega^6 \\
1 & \omega^3 & \omega^6 & \omega & \omega^4 & \omega^7 & \omega^2 & \omega^5 \\
1 & \omega^4 & 1 & \omega^4 & 1 & \omega^4 & 1 & \omega^4 \\
1 & \omega^5 & \omega^2 & \omega^7 & \omega^4 & \omega & \omega^6 & \omega^3 \\
1 & \omega^6 & \omega^4 & \omega^2 & 1 & \omega^6 & \omega^4 & \omega^2 \\
1 & \omega^7 & \omega^6 & \omega^5 & \omega^4 & \omega^3 & \omega^2 & \omega \\
\end{pmatrix},
\]  (17.35)

\[
U_1^{(8)} = \frac{1}{\sqrt{2}} \begin{pmatrix}
1 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \\
0 & 1 & 0 & 0 & 0 & 1 & 0 & 0 \\
0 & 0 & 1 & 0 & 0 & 0 & 1 & 0 \\
0 & 0 & 0 & 1 & 0 & 0 & 0 & 1 \\
1 & 0 & 0 & 0 & \omega^4 & 0 & 0 & 0 \\
0 & 1 & 0 & 0 & 0 & \omega^4 & 0 & 0 \\
0 & 0 & 1 & 0 & 0 & 0 & \omega^4 & 0 \\
0 & 0 & 0 & 1 & 0 & 0 & 0 & \omega^4 \\
\end{pmatrix},
\]  (17.36)

\[
U_2^{(8)} = \frac{1}{\sqrt{2}} \begin{pmatrix}
1 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\
0 & 1 & 0 & 1 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 1 & 0 & \omega^2 & 0 \\
0 & 0 & 0 & 0 & 0 & 1 & 0 & \omega^2 \\
1 & 0 & \omega^4 & 0 & 0 & 0 & 0 & 0 \\
0 & 1 & \omega^4 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 1 & 0 & \omega^6 & 0 \\
0 & 0 & 0 & 0 & 0 & 1 & 0 & \omega^6 \\
\end{pmatrix},
\]  (17.37)

and

\[
U_3^{(8)} = \frac{1}{\sqrt{2}} \begin{pmatrix}
1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 1 & \omega & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 1 & \omega^2 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 1 & \omega^3 \\
1 & \omega^4 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 1 & \omega^5 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 1 & \omega^6 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 1 & \omega^7 \\
\end{pmatrix}.
\]  (17.38)

One can verify by doing the matrix multiplication (using Mathematica helps) that Eq. (17.34) is satisfied.

One can see from Fig. 17.4 that the QFT can be written as\footnote{Recall that we work from right to left in operator equations like Eq. (17.39) but from left to right in circuit diagrams such as Fig. 17.4.}

\[
U^{(8)} = S_{02}^{(8)} H_l^{(8)} R_{1,m}^{(8)} H_m^{(8)} R_{2,u}^{(8)} R_{1,u}^{(8)} H_u^{(8)},
\]  (17.39)
in a fairly obvious notation, where

\[
S^{(8)}_{02} = \begin{pmatrix}
1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \\
0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \\
0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \\
0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \\
\end{pmatrix},
\]  
\[17.40\]

\[
H^{(8)}_l = \frac{1}{\sqrt{2}} \begin{pmatrix}
1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\
1 & -1 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 1 & 1 & 0 & 0 & 0 & 0 \\
0 & 0 & 1 & -1 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 1 & 1 & 0 & 0 \\
0 & 0 & 0 & 0 & 1 & -1 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 1 & 1 \\
0 & 0 & 0 & 0 & 0 & 0 & 1 & -1 \\
\end{pmatrix},
\]  
\[17.41\]

\[
R^{(8)}_{1,m} = \begin{pmatrix}
1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & i & 0 & 0 & 0 & 0 \\
0 & 1 & 0 & 0 & 1 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & i & 0 \\
\end{pmatrix},
\]  
\[17.42\]

\[
H^{(8)}_m = \frac{1}{\sqrt{2}} \begin{pmatrix}
1 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\
0 & 1 & 0 & 1 & 0 & 0 & 0 & 0 \\
1 & 0 & -1 & 0 & 0 & 0 & 0 & 0 \\
0 & 1 & 0 & -1 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 1 & 0 & 1 & 0 \\
0 & 0 & 0 & 0 & 0 & 1 & 0 & 1 \\
0 & 0 & 0 & 0 & 1 & 0 & -1 & 0 \\
0 & 0 & 0 & 0 & 0 & 1 & 0 & -1 \\
\end{pmatrix},
\]  
\[17.43\]

\[
R^{(8)}_{2,u} = \begin{pmatrix}
1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \\
0 & 1 & 0 & 0 & 1 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & \omega & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & \omega \\
\end{pmatrix},
\]  
\[17.44\]
CHAPTER 17. THE QUANTUM FOURIER TRANSFORM (QFT) AND A COMPARISON WITH THE FFT

\[ R_{1,u}^{(8)} = \begin{pmatrix}
1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 1 & 0 \\
0 & 0 & 0 & 0 & 0 & 1 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & i \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & i
\end{pmatrix}, \tag{17.45} \]

\[ H_u^{(8)} = \frac{1}{\sqrt{2}} \begin{pmatrix}
1 & 0 & 0 & 0 & 1 & 0 & 0 \\
0 & 1 & 0 & 0 & 0 & 1 & 0 \\
0 & 0 & 1 & 0 & 0 & 0 & 1 \\
0 & 0 & 0 & 1 & 0 & 0 & 0 \\
1 & 0 & 0 & 0 & -1 & 0 & 0 \\
0 & 1 & 0 & 0 & 0 & -1 & 0 \\
0 & 0 & 1 & 0 & 1 & 0 & -1 \\
0 & 0 & 0 & 1 & 0 & 1 & 0 & -1
\end{pmatrix}, \tag{17.46} \]

One may verify Eq. (17.39) using \textit{Mathematica}. Note that \( S_{02} \) swaps qubits 0 and 2, as required to reverse the order of the qubits.

Can we make a connection between the individual matrices, \( U_1^{(8)}, U_2^{(8)}, \) and \( U_3^{(8)} \), in the FFT, Eq. (17.34), and the individual matrices, \( S_{02}, H_l^{(8)}, R_{1,m}^{(8)}, R_{2,u}^{(8)}, R_{1,u}^{(8)}, \) and \( H_u^{(8)} \), in the QFT, Eq. (17.39)?

One immediately sees that \( U_1^{(8)} = H_u^{(8)} \). However to make a connection between the other parts of the FFT, \( U_2^{(8)} \) and \( U_3^{(8)} \), we introduce the swap operator between qubits 1 and 2:

\[ S_{12}^{(8)} = \begin{pmatrix}
1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \\
0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 1
\end{pmatrix}, \tag{17.47} \]

We also need to realize that we can move the \( R_2 \) gate in Fig. 17.4 to the right as long as it does not cross the Hadamard on the lowest qubit (since this is the control qubit). Hence we can also write Eq. (17.39) as

\[ U^{(8)} = S_{02}^{(8)} H_l^{(8)} R_{1,m}^{(8)} R_{2,u}^{(8)} H_m^{(8)} R_{1,u}^{(8)} H_u^{(8)}, \tag{17.48} \]

where we have moved \( R_{2,u}^{(8)} \) to the left. We then find that

\[ U_1^{(8)} = H_u^{(8)}, \tag{17.49a} \]
\[ U_2^{(8)} = S_{12}^{(8)} H_m^{(8)} R_{1,u}^{(8)}, \tag{17.49b} \]
\[ U_3^{(8)} = S_{02}^{(8)} H_l^{(8)} R_{1,m}^{(8)} R_{2,u}^{(8)} S_{12}^{(8)}, \tag{17.49c} \]

which agrees with Eqs. (17.38) and (17.34) since \( (S_{12}^{(8)})^2 \) is the identity (swapping twice makes no change). This breakup is shown in Fig. 17.7. Apart from the reversals of qubit order, the correspondence between the QFT and the FFT is straightforward to see.
17.C. COMPARISON OF FFT AND QFT FOR $N = 8$

Figure 17.7: Like Fig. 17.4 except that the $R_2$ gate has been moved to the right of the Hadamard on the middle qubit (which has no effect) and that a pair of reversals of the order of qubits 1 and 2 have been added (which also has no effect). The reversal is accomplished by a swap gate. Note that the final reversal of the order of all three qubits (on the right of the diagram) is also accomplished by a single swap gate. The correspondence with the breakup of the FFT ($U = U_3U_2U_1$) is indicated, see Eqs. (17.49). To see this correspondence it is necessary to include the pair of reversals of the order of qubits 1 and 2.

Figure 17.8: The generalization of Figs. 17.7 and 17.6 to the case of four qubits. The correspondence with the breakup of the FFT ($U = U_4U_3U_2U_1$) is indicated.

Following the structure of Fig. 17.6 for two qubits, and Fig. 17.7 for three qubits the generalization to four qubits is shown in Fig. 17.8. The correspondence with the FFT is clear, the only complication being that, in order to show the correspondence, pairs of reversals of the order of the qubits (which
cancel each other out) have to be introduced, with one reversal being in one stage of the FFT and the other reversal in the next stage of the FFT. Reading Fig. 17.8 from left to right, the first reversal pair reverses qubits 2 and 3 (which needs a single swap gate between qubits 2 and 3), the next reversal reverses qubits 1, 2 and 3 (which only needs a single swap gate between qubits 1 and 3), and the last reversal (not a pair because this is the last one so there is no additional stage to compensate it) reverses all 4 qubits (which needs two swap gates, one between qubits 0 and 3 and the other between qubits 1 and 2).

We see that there is a close parallel between the breakup of the FFT and circuit of the QFT. The details are slightly complicated because one needs reversals of the order of the qubits to make the correspondence precise. Note that Fig. 1 in https://arxiv.org/pdf/1005.3730.pdf is related to the results presented here.
Chapter 18

Shor’s Algorithm

When computers we build become quantum,
Then spies of all factions will want ‘em.
Our codes will all fail,
And they’ll hack our email,
But crypto that’s quantum will daunt ‘em.

This is a slightly modified version of a limerick by Peter and Jennifer Shor. (The original version is printed in the book by Nielsen and Chuang [NC00].) Continuing in a literary vein, on p. 453 of Nielsen and Chuang is a very well-crafted (Shakespearean) sonnet by Daniel Gottesman on quantum error correction. It seems that quantum computing brings out latent literary qualities in scientists who work on it (except me!).

18.1 Introduction

Consider an integer \( N \) composed of two prime factors \( p \) and \( q \), i.e. \( N = pq \). In Chapter 15 we showed how to determine the factors of \( N \) from the period \( r \) of the function

\[
f(x) \equiv a^x \pmod{N},
\]

where \( a \) is some number less than \( N \) and which has no factors in common with \( N \). Since \( a^0 = 1 \), the period is the smallest value \( x = r \) such that

\[
a^r \pmod{N} = 1.
\]

In 1994 Peter Shor [Sho94] developed a famous quantum algorithm for period finding which is much more efficient for factoring large integers than any known algorithm running on a classical computer. The ability to factor a large integer can be used to decode messages sent down a public channel (such as the internet) which have been encrypted with the RSA scheme. The first four lines of the above limerick refer to this. The RSA encryption scheme is described in Chapter 14.

Here we describe in detail Shor’s algorithm to determine the period of the function \( f(x) \) in Eq. (18.1). Useful references are [Mer07, NC00, Vat16].

We denote by \( n_0 \) the number of bits needed to contain \( N \). In cryptography, \( N \) may have of order 600 digits (so \( n_0 \sim 2000 \) bits).

18.2 Modular Exponentiation

In Shor’s algorithm the period is found by a Quantum Fourier transform of the function in Eq. (18.1) evaluated for \( x = 0, 1, 2, \cdots, 2^n - 1 \). What do we take for \( n \)? Now the period may be comparable to

\[1\] The last line of the limerick refers to quantum key distribution (QKD) which will be discussed in Chapter 21.
N and according to Mermin [Mer07] we need at least $N$ periods in the data, i.e. $2^n > N^2$, and so set $n = 2n_0$. We will see why the doubling of the number of qubits is necessary in Sec. 18.5. Hence, if $n_0 \sim 2000$ we have $n \sim 4000$.

It would seem to be a formidable (nay, impossible) task to calculate $a^x \pmod{N}$ for $x$ from 1 to $2^{4000}$. However, it can be done as follows using quantum parallelism. Compute $a, a^2, a^4, \ldots, a^{2^n} \pmod{N}$ by successively squaring. This only takes $n$ multiplications and so is very quickly done on a classical or quantum computer. Let the binary expansion of $x$ be

$$x = x_{n-1}x_{n-2}\cdots x_2x_1x_0.$$  

Then we have

$$a^x = \prod_{j=0}^{n-1} (a^{2^j})^{x_j}.$$  

For example for $n = 4$, $x = 10$, the binary expansion of $x$ is 1010 (note the least significant bit is to the right) so

$$a^x = (a^8)^1(a^4)^0(a^2)^1(a^1)^0.$$  

On a quantum computer each digit of $x$ is represented by one qubit, and, as we shall see, Eq. (18.4) can be computed efficiently for all $x$ between 1 and $2^n - 1$ on a quantum computer using quantum parallelism. The use of Eq. (18.4) to compute $a^x$ for a huge range of $x$ is called “modular exponentiation”.

A schematic circuit diagram for doing modular exponentiation is shown in Fig. 18.1. There are $n$ upper or “input” qubits and $n_0$ lower or “output” qubits, where one usually takes $n = 2n_0$ as discussed above. We will call the set of input qubits the “input register”, and similarly denote the output qubits as the “output register”. The notation “input” and “output” can be rather confusing. It is used because, after modular exponentiation, the input qubits contain the values of $x$, and the output qubits contain the function values $f(x)$. Nonetheless the both input and output registers are present at the start of the process (left edge of the circuit diagram in Fig. 18.1) and at the end (right edge of the circuit).

![Circuit Diagram](image)

Figure 18.1: Schematic circuit diagram for performing the modular exponentiation. The workings of the black box $U$ are described in the text. The state entering $U$ is $|\psi_0\rangle = \frac{1}{2^{n/2}} \sum_{x=0}^{2^n-1} |x\rangle_n |0\rangle_{n_0}$ and the state exiting from $U$ is $|\psi_1\rangle = \frac{1}{2^{n/2}} \sum_{x=0}^{2^n-1} |x\rangle_n |f(x)\rangle_{n_0}$.

Both the input and output qubits are initialized to $|0\rangle$. The input qubits are each run through a Hadamard gate. A Hadamard gate acting on state $|0\rangle$ of one qubit gives the symmetric combination $\frac{1}{\sqrt{2}}(|0\rangle + |1\rangle)$. Hadamards acting on $n$ qubits gives the symmetric sum of all $2^n$ basis states. Hence before entering into the box $U$ shown in Fig. 18.1 the state of the system is

$$|\psi_0\rangle = \frac{1}{2^{n/2}} \sum_{x=0}^{2^n-1} |x\rangle_n |0\rangle_{n_0}.$$  

(18.6)
On exiting the box $U$, the state of the system has the values of $f(x)$ in the output register, i.e.

$$|\psi_1\rangle = \frac{1}{2^{n/2}} \sum_{x=0}^{2^n-1} |x\rangle_n |f(x)\rangle_{n_0}. \quad (18.7)$$

Note that, in general, if the lower (output) qubits were initialized to $|y\rangle$, then after the function acted they would be in state $|y \oplus f(x)\rangle$, but here $y = 0$.

How does the function-evaluating box $U$ work? We start with a certain value for $x \equiv x_{n-1}x_{n-2} \cdots x_2x_1x_0$ in the input register and $1 \equiv 000 \cdots 001$ in the output register. We also need an additional work register with $n_0$ qubits, whose contents we will denote by $w$, with initial value $w = a$. The following steps compute $a^x \pmod{N}$ using Eq. (18.4):

- (a) Multiply the output register by $w$ if $x_0 = 1$.
- (b) Replace $w$ by its square $w \rightarrow w^2$.
- (a') Repeat (a) but for $x_1$.
- (b') Repeat (b).
- Continue repeating (a) (with successive bits of $x$) and (b).

Since all possible values of $x$ occur in the linear superposition in $|\psi_0\rangle$ in Eq. (18.6), (which is inputted to $U$) linearity of the operations in $U$ ensures that the state outputted by $U$ is the linear superposition in Eq. (18.7), with $f(x)$ computed for all $x$.

How many operations does this require? If we consider (b) we need to do $n$ squares of an $n_0$-bit number. Multiplying two $n_0$ bit numbers in the simplest way takes $O(n_0^2)$ operations. Since $n = 2n_0$ we see that the operation count for (b) is $O(n^3)$. The operation count for (a) is similar, so the total operation count for modular exponentiation is $O(n^3)$.

### 18.3 Quantum Fourier Transform (QFT)

A schematic of the full circuit for period finding is shown in Fig. 18.2.

The first (left) part of the algorithm is the modular exponentiation also shown in Fig. 18.1. A measurement is then made of the result in the “output” register from the modular exponentiation routine $U$. This is indicated by the box with arrow in Fig. 18.2. The measurement will yield some value for $f(x)$, say $f_0$. The input register will then contain a superposition of those basis states for which $f(x) = f_0$. Since $f(x)$ is periodic with period $r$, the possible values of $x$ are of the form $x_0 + kr$, so, after the measurement on the output register, the state of the input register becomes

$$|\psi_2\rangle = \frac{1}{\sqrt{Q}} \sum_{k=0}^{Q-1} |x_0 + kr\rangle_n. \quad (18.8)$$

Here $x_0 < r$, $x_0 + kr < 2^n$ and the number states in the sum is

$$Q = \left\lceil \frac{2^n}{r} \right\rceil, \quad (18.9)$$

where $\lceil \cdot \rceil$ denotes the integer part. Thus $P_x(x)$, the probability of $x$, consists of $Q$ delta functions at positions $x_0 + kr$, $k = 0, 1, \cdots, Q-1$, see Fig. 18.3.

---

As mentioned in Nielsen and Chuang [NC00], there are more sophisticated methods of multiplying $n$-bit numbers which only take $n (\ln n)$ (ln ln $n$) operations rather than $n^2$. This gives a total operation count for modular exponentiation of $O(n^2 \ln n \ln \ln n)$, hardly more than $O(n^3)$.
Figure 18.2: Schematic circuit diagram for Shor’s algorithm for period finding on a quantum computer. The black box $U$ does the modular exponentiation as described in the text, see also Fig. 18.1. The state inputted to $U$ is given by $|\psi_0\rangle$ in Eq. (18.6) and the state outputted from $U$ is given by $|\psi_1\rangle$ in Eq. (18.7). A measurement (indicated by the box with the arrow) is performed on the output register, giving some value $f_0$. The double lines indicate that the measurement gives classical bits which take values 0 or 1. The state of the input register is then given by $|\psi_2\rangle$ in Eq. (18.8), the equally weighted superposition of all values of $x$ for which $f(x) = f_0$. The $n$ input qubits then go through the quantum Fourier transform the result of which is given by $|\psi_3\rangle$ in Eq. (18.10). A measurement of the input qubits then gives a result $y$ which is close to an integer multiple of $2^n/r$, where $r$ is the period, as discussed in the text.

Figure 18.3: The probability of getting state $x$ in the input register if a measurement were performed before doing the Quantum Fourier Transform. There are $Q$ delta functions, each with weight $1/Q$ separated by $r$, the period. The values of $x$ where these delta functions appear, $x_0 + kr, k = 0, 1, \ldots, Q-1$, are those values for which $f(x) = f_0$ the result obtained from the measurement of the output register. A measurement would get a value for $x_0 + kr$ for some $k$ but since we don’t know $x_0$ this is no help in determining the period $r$. Hence measuring the input register at this point is not useful. We need to Fourier transform the state in the input register before measuring it, in order to determine the period.
If we were to measure $|\psi_2\rangle$ we would just get one value of $x_0 + kr$, which, because of the dependence on the unknown quantity $x_0$, does not give any information from which we might be able to determine the period $r$. Therefore, in order to extract information on $r$, we perform a quantum Fourier transform on the states in Eq. (18.8):

$$|\psi_3\rangle = \sum_{y=0}^{2^n-1} \left( \frac{1}{\sqrt{2^n}} \sum_{k=0}^{Q-1} e^{2\pi i (x_0 + kr)y/2^n} |y\rangle \right).$$  

(18.10)

The quantum circuit which performs the Quantum Fourier Transform is described in Chapter 17. An example for $n = 4$ qubits is shown in Fig. 18.4 in which the controlled phase gates act on the target qubit according to

$$R_d = \begin{pmatrix} 1 & 0 \\ 0 & e^{\pi i/2^d} \end{pmatrix}.$$  

(18.11)

if the control qubit is 1, and otherwise do nothing. Note that $R_0$ is just the Z gate. Like the controlled Z gate, the controlled phase gate is symmetric between the control and target qubits (the phase is changed only if both qubits are $|1\rangle$), so the control and target qubits can be exchanged. We will use this in Appendix 18.A when we see how to actually eliminate these 2-qubit gates.

Generalizing the diagram in Fig. 18.4 to the case of $n$ qubits we see that controlled phase gates $R_d$ are required for $d = 1, 2, \cdots, n - 1$. Hence, in total, we need $n$ Hadamard gates and $1 + 2 + \cdots + n - 1 = n(n-1)/2$ controlled phase gates. However, as discussed in Sec. 3.9 of Mermin [Mer07], and in Appendix 18.C, it is both impossible to construct gates giving a phase change which is exponentially small in $n$, and also not necessary to do this to obtain the QFT with the required precision. Mermin shows that one only needs controlled phase gates $R_d$ for $d < \log_2(\text{const. } n)$, where the constant Mermin gives is large but independent of $n$. Thus the number of controlled phase gates needed in practice is of order $n \log_2 n$ which is considerably less than $O(n^2)$ if $n$ is order order several thousand.

Figure 18.4: The circuit for the quantum Fourier transform for $n = 4$ qubits. The controlled phase gates act on the target qubit according to Eq. (18.11) if the control qubit is 1 and otherwise does nothing. The final swap gates to reverse the order of the qubits outputted on the right are not included here. Note that the controlled phase gate between qubits $x_i$ and $x_j$ is $R_{|i-j|}$ which makes the structure of the circuit quite simple to understand.
In fact we can eliminate the 2-qubit controlled phase gates by measuring each qubit immediately after the gates of the QFT have acted on it, rather than after completion of the QFT. This is discussed in Appendix 18.A.

After the quantum Fourier transform we measure the input register, see Fig. 18.2, obtaining a value for \( y \). The probability of getting a particular state \( y \) is given by the square of the absolute value of a term in the brackets in Eq. (18.10), i.e.

\[
P(y) = \frac{1}{2^n Q} \left| \sum_{k=0}^{Q-1} e^{2\pi i k y/2^n} \right|^2.
\]

(18.12)

Note that the dependence on \( x_0 \), which was troublesome before doing the Fourier transform, and appears just as a phase factor in the Fourier transform, Eq. (18.10), now drops out completely when we take the square of the absolute value to get the probabilities in Eq. (18.12).

If \( y \) could take real values, the exponentials would add up precisely in phase (and so there would be a peak in the probability for \( y \)), when \( yr/2^n \) is an integer, i.e. for \( y = y_m \) where

\[
y_m = m \frac{2^n}{r},
\]

(18.13)

in which \( m \) is an integer. Note that there are \( r \) values of \( m \), from 0 to \( r - 1 \) since \( y \) runs over a range of \( 2^n \) values. We emphasize that \( y_m \) is not an integer in general, but the measured values of \( y \) are integers, so there will be peaks in \( P(y) \) at integer values close to the \( y_m \) in Eq. (18.13), see the sketch in Fig. 18.5. Precise values of \( P(y) \) for a particular case will be calculated in Sec. 18.5. Hence there is a high probability that we will obtain an integer close to an integral multiple of \( 2^n/r \).

![Figure 18.5](image-url)

Figure 18.5: A sketch of the probability of getting state \( y \) in the input register after the Quantum Fourier Transform. There are \( r \) peaks at \( y_m = m \frac{2^n}{r} \) for \( m = 0, 1, 2, \cdots , r - 1 \). Note that \( \lceil 2^n/r \rceil = Q \) so the separation between the peaks in \( P(y) \) is, no more than 1 away from \( Q \), the number of peaks in the distribution \( P_x(x) \) for the state before the quantum Fourier transform, see Fig. 18.3. Precise values of \( P(y) \) will be calculated in Sec. 18.5 for a particular case and the resulting values of \( P(y) \) will be shown in Fig. 18.8.

To summarize this part, \( P(y) \) has \( r \) peaks separated by \( 2^n/r \) (the nearest integer below this is \( Q \)), see Fig. 18.5. Note that \( r \), the number of peaks after the QFT (Fig. 18.5), is equal to the separation...

Between the peaks before the QFT (Fig. 18.3) and the separation between the peaks after the QFT (essentially Q) is equal to the number of peaks before the QFT. This interchange of the number of peaks and their separation on doing the QFT is characteristic of the discrete Fourier Transform.

18.4 A special case: the period r is a power of 2.

In some special cases the period r will be a power of 2. An example discussed by Mermin [Mer07] is if both p and q are both primes of the form $2^\ell + 1$ (e.g. the commonly studied case of $N = 15$). When the period is a power of 2, it can be shown that the range of $x$ (i.e. $2^n$) will contain an exact integer number of periods. In this situation we will not need $n$ to be as big as $2^{n_0}$ (where $n_0$ is the number of bits needed to contain N). Rather, we will see that we just need $2^n$ to be big enough to contain some integer number of periods for us to exactly determine an integer multiple of $2^n/r$. Since the period might be as large as N, when $r$ is a power of 2 we need

$$2^n = \text{const. } 2^{n_0} \quad \text{rather than}$$

$$2^n = 2^{2n_0} \quad \text{in the general case}. \quad (18.14)$$

Here we go through this special case because the mathematics is simpler than the generic case which we will study in the next section.

First of all we check for $N = 15$ that the period is a power of 2 as stated above. Let’s take $a = 7$ which has no factors in common with 15:

$$x = 1, \quad a^x = 7,$$

$$x = 2, \quad a^x = 7 \times 7 = 49 \equiv 4 \pmod{15}, \quad (18.15a)$$

$$x = 3, \quad a^x = 7 \times 4 = 28 \equiv 13 \pmod{15}, \quad (18.15b)$$

$$x = 4, \quad a^x = 7 \times 13 = 91 \equiv 1 \pmod{15}, \quad (18.15c)$$

so the period is $r = 4$, i.e. a power of 2 as claimed.

Now, we perform the sum in Eq. (18.12). We take any $n$ such that $2^n$ is a multiple of $r$. The value of $Q$ is given exactly by

$$Q = \frac{2^n}{r}, \quad (18.16)$$

so Eq. (18.12) becomes

$$P(y) = \frac{1}{2^n Q} \left| \sum_{k=0}^{Q-1} e^{2\pi i k y/2^n} \right|^2,$$

$$= \frac{1}{r} \left| \frac{1}{Q} \sum_{k=0}^{Q-1} e^{2\pi i k y/Q} \right|^2. \quad (18.17)$$

Firstly suppose that $y = mQ$ for integer $m$. It is trivial to see that all the exponentials in Eq. (18.17) are unity so

$$P(y = mQ) = \frac{1}{r}. \quad (18.18)$$

Note that there are $r$ distinct values of $m$, $m = 0, 1, 2, \cdots, r - 1$ since $y$ runs over a range of $2^n$ values and $Q = 2^n/r$, see Eq. (18.16). Hence the sum of the probabilities for these values of $y$ is unity. Since the total probability must be unity there can be no probability for other values of $y$, as we will now verify.
The sum in Eq. (18.17) is a geometric series, and for \( y \neq mQ \) it can be summed to give

\[
\sum_{k=0}^{Q-1} e^{2\pi i ky/Q} = \frac{1 - e^{2\pi iy}}{1 - e^{2\pi iy/Q}}. \tag{18.19}
\]

The numerator is zero for all \( y \) (recall that \( y \) is an integer), but for \( y \neq mQ \) the denominator is non-zero, so

\[ P(y \neq mQ) = 0, \tag{18.20} \]

as required. Thus, with probability 1, the measured value of \( y \) is an integer multiple of \( 2^n/r \). This is shown in Fig. 18.6. Superficially, this may look similar to the situation before the QFT shown in Fig. 18.3. The difference is that the known quantity \( x_0 \) does not appear in Fig. 18.6 and the delta functions occur at positions \( y_m \) where \( y_m/2^n = m/r \) from which one can determine \( r \). By contrast, the delta functions in Fig. 18.3 are at \( x_k = x_0 + kr \) from which one can not determine \( r \) because one has no idea of the value of \( x_0 \).

Notice the reciprocal relation between the period \( r \) in the original data in Fig. 18.3 and the period in the Fourier transformed data which is the size of the dataset, \( 2^n \), divided by \( r \). To use terminology from sound waves and frequencies, quite generally, if an original dataset is periodic, the Fourier transform will have a peak at the “fundamental frequency”, in this case \( 2^n/r \), and in addition can have peaks at “higher harmonics” (in this case \( m \cdot 2^n/r \) for \( m > 1 \)), and can also have a component at zero “frequency” (\( y = 0 \) here) if the average of the original data is non-zero. The special nature of the original dataset here (uniformly spaced delta functions, see Fig. 18.3) leads to the Fourier transform having equal weight in all the Fourier components.

Figure 18.6: The probability of getting state \( y \) in the input register after the Quantum Fourier Transform for the special case where \( r \) is a power of 2 so there are an exact number of periods in the interval \( 2^m \). There are \( Q = 2^n/r \) delta functions of equal weight at exactly \( y_m = m \cdot 2^n/r \), for \( m = 0, 1, \cdots, Q - 1 \).

Let us give a simple example so we can see in detail how to extract the period \( r \) from this knowledge. We take our previous example of \( N = 15, a = 7 \), for which we found in Eq. (18.15) that the period is \( r = 4 \). This means that \( 7^4 \equiv 1 \mod 15 \). We will assume that we have \( n = 5 \) qubits, so \( 2^n = 32 \). The only possible results of a measurement of \( y \) are an integer multiple of \( Q = 2^n/r \), so here we have \( y = 0, 8, 16 \) and 24, each with equal probability \( 1/4 \), see Table 18.1.
Table 18.1: The possible results of a measurement of $y$ for the case of $N = 15, a = 7, 2^n = 32$ for which $r = 4$. A measurement gives both the numerator and denominator of the fraction $y/2^n = (m_0/r_0)$, so we write $y/2^n$ with $m = cm_0, r = cr_0$. Hence we obtain $r_0$ (and $m_0$), but not $c$. We determine $c$ by trial and error, as discussed in the text.

<table>
<thead>
<tr>
<th>$y$</th>
<th>$m$</th>
<th>$y/2^n = (m_0/r_0)$</th>
<th>$c = r/r_0$</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>0</td>
<td>0</td>
<td>–</td>
</tr>
<tr>
<td>8</td>
<td>1</td>
<td>1/4</td>
<td>1</td>
</tr>
<tr>
<td>16</td>
<td>2</td>
<td>1/2</td>
<td>2</td>
</tr>
<tr>
<td>24</td>
<td>3</td>
<td>3/4</td>
<td>1</td>
</tr>
</tbody>
</table>

From the measurement of $y$ we determine the numerator and denominator of the fraction $y/2^n = (m/r)$ for some $m$, but with any common factor $c$, shown in the last column of Table 18.1 divided out, so we write $y/2^n$ with $m = cm_0, r = cr_0$. To determine $r = cr_0$, where $c$ is generally a small integer, we compute $a^{cr_0}$ mod $N$ for the first few values of $c = 1, 2, \cdots$ and see for what value of $c$ we obtain 1, the result if $cr_0 = r$, see Eq. (18.2). The common ratio $c$ is unlikely to be large. For example if $m$ is odd, which occurs with probability $1/2$, then $c = 1$. Similarly there is probability $1/4$ that $m$ is even but not a multiple of 4 in which case $c$ cannot be greater than 2. Proceeding in this vein we see that it is very unlikely that $c$ is large. In the rare case that the common ratio $c$ is large, we would stop after the first few values of $c$ and restart the quantum computation (the steps shown in Fig. 18.2).

In Table 18.1 we see that the value $y = 0$ does not give useful information but, since the number of possible results is equal to $r$ and each result is equally probable, the probability of getting $y = 0$ is small if the period $r$ is large (the usual situation if one needs a quantum computer).

In this section, we have seen that in the rare situation that the period is a power of 2, the measurement of $y$ gives an integer multiple of $2^n/r$ with probability one. Hence $y/2^n = m/r$ with integer $m$ exactly and there is no need to use the continued fraction method described in Appendix 18.B to determine $m/r$.

However, in the general case, which we discuss in the next section, we firstly need to have $n = 2n_0$ so there are at least $N$ periods in the range of $x$, and secondly the measurement of $y$ will give, with a probability which is high but less than one, a value such that $y/2^n$ is close to (but not equal to) $m/r$. The continued fraction method in Appendix 18.B is then needed to determine $m/r$.

18.5 The generic case: the period is not a power of 2.

We now evaluate the sum in Eq. (18.12) for the generic case when $r$ is not a power of 2 so we do not have an exact integer number of periods in the range of $x$-values, $2^n$, over which $f(x)$ is calculated. As discussed after Eq. (18.12), $P(y)$ has $r$ peaks in the vicinity of $y_m$ given by Eq. (18.13). We set

$$y = y_m + \delta_m = m_0 \frac{2^n}{r} + \delta_m,$$

(18.21)

We will assume that $\delta_m$ is small, so we are close to the $m$-th peak, but $2^n, r$ and $m$ are large, since we only need the quantum algorithm when these numbers are large. (Recall that $y$, the measured value is an integer, whereas $y_m$ and $\delta_m$ are not.)
Equation (18.12) involves a geometric series which can be summed as follows:

\[
\sum_{k=0}^{Q-1} e^{2\pi ikry/2^n} = \sum_{k=0}^{Q-1} e^{2\pi ikm} e^{2\pi ikr\delta_m/2^n},
\]

\[
= \sum_{k=0}^{Q-1} e^{2\pi ikr\delta_m/2^n},
\]

\[
= \frac{1 - e^{2\pi iQr\delta_m/2^n}}{1 - e^{2\pi ir\delta_m/2^n}},
\]

\[
= e^{\pi iQr\delta_m/2^n} \sin \left( \frac{\pi Qr\delta_m}{2^n} \right) e^{\pi ir\delta_m/2^n} \sin \left( \frac{\pi r\delta_m}{2^n} \right),
\]

(18.22)

Inserting Eq. (18.22) into Eq. (18.12) the phase factors drop out and we get

\[
P(y) = 1 - \frac{\sin^2 \left( \frac{\pi Qr\delta_m}{2^n} \right)}{\sin^2 \left( \frac{\pi r\delta_m}{2^n} \right)}.
\]

(18.23)

Now \( Q \) is within an integer of \( 2^n/r \) and \( Q \) is also large so we can replace \( Qr/2^n \) by 1 with negligible error. Also \( r/2^n \) is very small, since we take \( n \) to be big enough that there are many periods within the range of \( x \) computed, so the sine in the denominator can be replaced by its argument. Hence we get, to a good approximation,

\[
P(y) = \frac{1}{r} \left( \frac{\sin \pi \delta_m}{\pi \delta_m} \right)^2,
\]

(18.24)

for \( y \) in the vicinity of \( y_m \). Recall that the relation between \( \delta_m \) and \( y \) is given in Eq. (18.21). The function in Eq. (18.24) is plotted in Fig. 18.7. The area under the curve is 1, and most of the weight is in the peak centered at 0.

To find the period we would like to get the integer \( y \) which is closest to \( m 2^n/r \) for some integer \( m \) i.e. \( |\delta_m| < 1/2 \). Writing \( \pi \delta_m = x \), this corresponds to \( |x| < \pi/2 \), and in this region

\[
\frac{\sin x}{x} > \frac{2}{\pi},
\]

(18.25)

so, according to Eq. (18.24), the probability of getting the nearest integer to \( y_m \) is greater than

\[
\frac{1}{r} \frac{4}{\pi^2} \approx 0.40 \frac{r}{r},
\]

(18.26)

see Fig. 18.7. There are \( r \) distinct values of \( m \) so the total probability of getting the closest integer to one of the \( y_m \) is greater than 40\%.\footnote{One of these is for \( m = 0 \) which doesn’t give useful information but since we are interested in situations where \( r \) is large, the difference between \( r \) and \( r - 1 \) is negligible.}

So, with fairly high probability, we have obtained the nearest integer to \( m 2^n/r \) for some integer \( m \) (which we don’t know). How can we determine \( r \) from this information? We need some post-processing which will be done on a classical computer.

In deriving Eq. (18.24) we just needed that the range of \( x \) studied contains many periods, i.e. \( 2^n \gg r \). Since \( r \) can not be bigger than \( N \) we needed \( 2^n \gg N \). However, to actually extract \( r \) we need a stronger condition, \( 2^n > N^2 \), as we shall now see.

\footnote{In fact, according to Mermin [Mer07], Appendix L, when \( N \) is the product of two primes (as we have here) the period is not only less than \( N \) but less than \( N/2 \). As a result, still using \( n = 2m_0 \) qubits in the input register, the algorithm will provide a divisor of \( r \) not only if the measured value of \( y \) is the closest integer to \( m 2^n/r \), but also if it is the second, third or fourth closest. This increases the probability of a successful run to about 0.9.}
18.5. THE GENERIC CASE: THE PERIOD IS NOT A POWER OF 2.

![Figure 18.7: A plot of the function in Eq. (18.24), neglecting the factor of $1/r$ where $r$ is the number of peaks. The area under the curve is 1. The result of a measurement will be one of a series uniformly spaced possible values of $\delta_m$ separated by 1, e.g. $\cdots, -1.7, -0.7, 0.3, 1.3, \cdots$. An example of real data is shown in Fig. 18.9. One of these values for $\delta_m$ must be within $1/2$ of 0 and the figure shows that the probability for this is greater than $4/\pi^2$, the dashed horizontal line. (The dashed vertical lines are at $\delta_m = \pm 1/2$).

We assume now that we have been successful and found a $y$ which is within $1/2$ of $2^n m/r$. Dividing by $2^n$ we have

$$\left| \frac{y}{2^n} - \frac{m}{r} \right| < \frac{1}{2^{n+1}},$$

so $y/2^n$, our estimate for $m/r$, is off by no more than $1/(2^{2n})$. The value of $m/r$ can then be obtained using continued fractions, which are discussed in Appendix 18.B. The crucial result of continued fractions which we need is theorem A4.16 in Appendix 4 of Ref. [NC00], which states that if

$$\left| \frac{y}{2^n} - \frac{m}{r} \right| < \frac{1}{2^{n+1}}, $$

then $m/r$ is one of the partial sums in the continued fraction representation of $y/2^n$. Here $r < N \sim 2^{n_0} = 2^{n/2}$ so we see from Eq. (18.27) that the theorem applies. Hence $m/r$ will appear as one of the partial sums in the continued fraction representation of $y/2^n$. Since $r < N$ this must be a partial sum with denominator less than $N$. Successive partial sums get more and more accurate, so we want the one with the largest denominator less than $N$.

If $m$ and $r$ have a common factor, $c$ say, then the continued fraction representation will divide this out and give $m_0/r_0$ where $m_0 = m/c, r_0 = r/c$. Thus we actually get $r_0$ which is a divisor of $r$. However, we may be lucky and still get $r$ straight away. As shown in Appendix J of Mermin [Mer07],

If we have two approximants for $y/2^n$, $p/q$ and $p'/q'$ say, then

$$\left| \frac{p}{q} - \frac{p'}{q'} \right| = \frac{|pq' - p'q|}{qq'} > \frac{1}{N^2},$$

(since $q$ and $q'$ are less than $N$) unless the two approximants are equal, so there is at most one approximant with denominator less than $N$ which satisfies Eq. (18.27).
the probability that two large numbers chosen at random have no common factors is greater than 1/2. Thus, with probability greater than 1/2, we get \( r \) directly. We can check if \( r = 0 \) is the period \( r \) by computing, on a classical computer, \( a^{r_0}\pmod{N} \) and seeing if we get 1. If we do not, we would try simple multiples, \( r = 2r_0, 3r_0, 4r_0, \ldots \), since it is very unlikely that the common factor is large. If we are very unlucky, and the common factor is large, we could start again from the beginning, get another value for \( m/r \) and hence get another value for \( r_0 \), and compute \( a^{r_0}\pmod{N} \). If this is not 1, then again we try \( r = 2r_0, 3r_0, 4r_0, \ldots \). There is also a chance that the measured value of \( y \) is not close enough to one of the \( y_m \) to get the period from continued fractions. Again, if this happens we need to repeat the whole procedure. However, we will not have to repeat very many times because the probability of success in one run is quite high.

The probabilistic nature of Shor’s algorithm, with the resultant need to run the algorithm several times (usually not very many), is a quite common feature of quantum algorithms.

18.6 An example

The last section was probably hard going, so we will try to clarify things by discussing a simple example. Consider the following, which was also discussed in Chapter 15, \( N = 91, a = 4 \). The period is \( r = 6 \) since

\[
\begin{align*}
  x &= 1, \quad a^x = 4, \\
  x &= 2, \quad a^x = 16, \\
  x &= 3, \quad a^x = 64, \\
  x &= 4, \quad a^x = 64 \times 4 = 256 = 2 \times 91 + 74 \equiv 74 \pmod{91}, \\
  x &= 5, \quad a^x = 74 \times 4 = 296 = 3 \times 91 + 23 \equiv 23 \pmod{91}, \\
  x &= 6, \quad a^x = 23 \times 4 = 92 = 91 + 1 \equiv 1 \pmod{91}.
\end{align*}
\]

Since the period is not a power of 2 this is a generic example, as discussed in the previous two sections.

<table>
<thead>
<tr>
<th>order ((m))</th>
<th>peak position ((y_m = m 2^n/r))</th>
<th>nearest integer</th>
<th>(P) (nearest int.)</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0.167</td>
</tr>
<tr>
<td>1</td>
<td>2730.67</td>
<td>2731</td>
<td>0.114</td>
</tr>
<tr>
<td>2</td>
<td>5461.33</td>
<td>5461</td>
<td>0.114</td>
</tr>
<tr>
<td>3</td>
<td>8192</td>
<td>8192</td>
<td>0.167</td>
</tr>
<tr>
<td>4</td>
<td>10922.67</td>
<td>10923</td>
<td>0.114</td>
</tr>
<tr>
<td>5</td>
<td>13653.33</td>
<td>13653</td>
<td>0.114</td>
</tr>
</tbody>
</table>

Table 18.2: The peak positions in the Fourier transform for the example discussed in this chapter. The output is at integer values of \( y \) and the nearest integers to the peaks are shown along with the probability at those nearest integer values. Neglecting the zeroth order peak at \( y = 0 \), which doesn’t give useful information, the sum of the other probabilities at the nearest integers is 0.623, so we have a greater than 60% probability of obtaining the nearest integer to a non-zero multiple of \( 2^n/r \), from which one can deduce \( r \) using continued fractions, as discussed in the text and Appendix 18.B.

One needs \( n_0 = 7 \) bits to represent \( N \) so we take \( n = 2n_0 = 14 \). Hence

\[
\frac{2^n}{r} = 2730.67
\]

so

\[
Q = 2730.
\]
18.6. AN EXAMPLE

Hence there are 2730 (and two thirds) periods in our data. As discussed in Mermin [Mer07] and Sec. 18.5, we need at least \( N (= 91) \) periods so 2730 is something of an overkill. The peaks in the Fourier transform, which are at integers next to multiples of \( 2^n/r \) as discussed above, are shown in Table 18.2.

![Figure 18.8: Probabilities for the different components of the Fourier transformed state for the example studied with \( N = 91, a = 4 \) for which the period is \( r = 6 \). There are six sharp peaks near \( y_m = m \cdot 2^n/r \), for \( m = 0, 1, \ldots, 5 \). The one at \( y = 0 \) (\( m = 0 \)) doesn’t give useful information. However, the probability of hitting the highest point of one of the other five peaks, i.e. the nearest integer to a non-zero multiple of \( 2^n/r \), is greater than 60%, see Table 18.2. If, as is likely, the measurement gives one of these results, it can then be used to determine the period \( r \), as discussed in the text and Appendix 18.B. A blowup of the \( m = 2 \) peak is shown in Fig. 18.9.](image)

I have evaluated \( P(y) \) numerically from Eq. (18.12) and the results are shown in Fig. 18.8. There are \( r = 6 \) peaks at values close to \( y_m = m \cdot 2^n/r \). There is a trivial peak at exactly \( y = 0 \) (\( m = 0 \)) but this can not give any useful information about the period \( r \). For the other 5 peaks, the peaks are not, in general, centered at exactly integer values, so the possible observed (integer) values of \( y \) are a set of discrete values around each peak, as shown in the histogram in Fig. 18.9 which blows up the region around the \( m = 2 \) peak.

As discussed in Sec. 18.5, the sum in Eq. (18.12) can be evaluated, and is given, to a good approximation, by Eq. (18.24) in the region of the \( m \)-th peak, where \( y \) is given by Eq. (18.21), and \( y_m \), given by Eq. (18.13), indicates the peak position. (Recall that \( y \) itself is an integer.) The function in Eq. (18.24) is plotted for continuous \( y \) as the solid curve in Fig. 18.9. When evaluated at integer \( y \), it agrees very well with the values numerically computed from Eq. (18.12) which are shown as the histogram in Fig. 18.9.

Note that \( \delta_m \) in Eq. (18.24) is defined in Eq. (18.21) and can be written as

\[
\delta_m = \epsilon + \ell
\]  

(18.32)
where $\ell$ is an integer and $|\epsilon| < 0.5$. Note too that

$$\sum_{\ell=-\infty}^{\infty} \left( \frac{\sin(\pi(\epsilon + \ell))}{\pi(\epsilon + \ell)} \right)^2 = 1,$$

(18.33)

for arbitrary $\epsilon$ (recent versions of Mathematica know this). Hence, according to Eqs. (18.24), (18.32), and (18.33), the weight around each of the peaks in Fig. 18.8 is equal to $1/r (= 1/6$ here). There are $r$ peaks so the total probability is $r \times (1/r) = 1$ as required. Referring to Fig. 18.9, the weight in the largest bar is 0.114 which is 68% of 1/6, the total weight in all the bars for this peak.

From Table 18.2 we see that the probability of getting the nearest integer to an integral multiple of $2^n/r$ is greater than 60%. Let’s suppose we get one of these. In fact, lets suppose we get the large bar at $y = 5461$ in Fig. 18.9. (Recall that Fig. 18.9 is a blowup of the $m = 2$ peak in Fig. 18.8.) Given the measured value, $y = 5461$, we will now see how to determine the period $r$ using continued fractions which are described in Appendix 18.B. We define $x = y/2^n$. This is close to $m/r$, where $r$, the period, is what we want to determine.

Since $r$ is no greater than $N$, as discussed in Sec. 18.5, the best guess for $x$ is the partial sum having the largest denominator less than $N$. As stated above we assume measurement gives the value $y = 5461$, the highest histogram for the peak in Fig. 18.9. Then we determine the continued fraction representation for $x = 5461/16384$ (since $n = 14$ we have $2^n = 16384$). Since this is a rational fraction the continued fraction terminates and has only the following denominators

$$c_0 = 0, \ c_1 = 3, \ c_2 = 5461.$$  

(18.34)
The partial sums are 0, 1/3 and 5461/16384. The latter has a denominator bigger than \( N = 91 \) so we neglect it and conclude that
\[
\frac{m}{r} = \frac{1}{3}.
\] (18.35)

It is possible that \( m \) and \( r \) have a common factor, i.e. \( m = c, r = 3c \) for some integer \( c \). We try some small values for \( c \). Starting with \( c = 1 \), so \( r = 3 \), we compute \( a^3 \mod 91 \) and find that it is not 1, see Eq. (18.29c). However, we find that \( c = 2 \) does work, since \( a^6 \equiv 1 \mod 91 \), see Eq. (18.29f). Hence the period \( r \) is equal to 6, the desired result.

18.7 Summary

What is the operation count for Shor’s period finding algorithm?

To factor an integer with \( n \) bits, the QFT requires, in principle, \( O(n^2) \) operations, as shown in section 18.3. Note, however, as discussed there, in Appendix 18.C and in Mermin [Mer07], in practice one only needs of order \( n \log_2 n \) gates to do the QFT to the necessary precision.

The computation of the function values using modular exponentiation takes \( O(n^3) \) operations, as shown in section 18.2 (but see footnote 2 on page 127 which states that the operation count is \( O(n^2 \log n \log \log n) \), not much more than \( O(n^2) \), if one uses a sophisticated method for multiplying two large numbers).

What about the continued fraction part, which is, of course, done on a classical computer? Each division of an \( n \)-bit number takes of order \( n^2 \) operations if the division is done in a simple way. In fact, division can be rewritten as several multiplications, see https://en.wikipedia.org/wiki/Division_algorithm, so the operation count can be reduced to that for multiplication, i.e. \( O(n \log n \log \log n) \). The depth of the continued fraction where the denominator is \( O(N) \) is \( O(\log N) \), since the coefficients in the continued fraction multiply to get the numerator and denominator. This is \( O(n) \) since \( N \) contains no more than \( n/2 \) bits. Hence the operation count for the continued fraction post-processing is \( O(n^3) \), but recall that this is done on a classical computer. Again the count is not much more than \( O(n^2) \) if one uses a sophisticated method for dividing two large numbers.

Hence, the overall operation count of Shor’s algorithm is \( O(n^3) \), which can be reduced to \( O(n^2 \log n \log \log n) \) using sophisticated methods for multiplying and dividing large numbers.

Shor’s algorithm for factoring integers therefore runs in polynomial time as a function of \( n \), the number of bits in \( N \). For comparison, no polynomial time classical algorithm for factoring integers is known. The fastest known classical algorithm, the general number field sieve (GNFS), takes a time \( \exp(\text{const.} n^{1/3} \log^{2/3} n) \). It is currently not known whether a polynomial time classical algorithm exists or not.

Even though the power of \( n \) in the exponent of the GNFS algorithm is less than one, it still much slower for large \( n \) than Shor’s polynomial-time algorithm. Hence, if the considerable technical difficulties could be overcome, and a quantum computer with a sufficiently large number of qubits built with the error rate made sufficiently low, then such a device could decode encrypted messages currently being sent down the internet, and which are currently impossible to decode on a classical computer.

\footnote{In this case, where there are many more than \( N \) periods in the intervals 2\(^n\), one gets the right answer if the measurement gives one of the other nearby \( y \) values. For example, if we get \( y = 5760 \) (the third closest to the peak), the continued fraction coefficients are 0, 3, and 1365. We do not include 1365 because it gives a denominator greater than \( N \), so we again get \( m/r = 1/3 \).}
18.A Eliminating the two-qubit gates

It is possible to replace the 2-qubit gates by 1-qubit gates which act or not depending on the result of a measurement. This is important from a technological point of view since 1-qubit gates are much easier to implement than 2-qubit gates. The point is that we measure the final state of the QFT anyway, and we will see that we can eliminate the 2-qubit gates by measuring each qubit immediately all the gates of the QFT have acted on it rather than waiting until the QFT is completed. We now see how to do this.

![Figure 18.10: Circuit equivalent to Fig. 18.4 but with the target and control qubits interchanged on the controlled phase gates.](image)

First of all we note that, similar to the control-Z gate, the target and control qubits in the controlled phase gates can be interchanged. Hence Fig. 18.4 is equivalent to Fig. 18.10.

In Fig. 18.10 we see that, for each qubit, once the Hadamard and the phase gates have acted nothing else happens so the qubit could be measured at this point. (Recall that time flows from left to right in circuit diagrams). Consider the top qubit $x_3$ which, on output, is $y_0$. We measure it immediately after the Hadamard. If the result is $y_0 = 1$ then the $R_1$ phase gate for $x_2$ is activated, as well as the $R_2$ phase gate for $x_1$ and the $R_3$ phase gate for $x_0$. However, if the result is $y_0 = 0$ then those phase gates are not activated. Since $y_0$ has been measured, this control can be done by a classical circuit, which is much simpler to implement than a 2-qubit quantum gate. Similarly we measure $x_2$, which is $y_1$ on output, immediately after its Hadamard. Hence the $R_1$ gate on $x_1$ and the $R_2$ gate on $x_0$ can be activated classically if $y_1 = 1$. We can proceed in this way for the whole circuit, measuring the qubit after the Hadamard, and using the result to phase change other qubits, or not, using classical control. The circuit is shown in Fig. 18.11.

18.B Continued Fractions

Continued fractions are a convenient way of finding a simple rational approximation to a number.
The continued fraction representation of a number \( x \) is obtained as follows. If there is an integer part of \( x \) call this \( c_0 \). Subtract \( c_0 \) from \( x \) and call the inverse of the remainder \( x_1 \), so

\[
x = c_0 + \frac{1}{x_1}.
\]  

(18.36)

Let the integer part of \( x_1 \) be \( c_1 \). Subtract \( c_1 \) from \( x_1 \) and call the inverse of the remainder \( x_2 \) so

\[
x_1 = c_1 + \frac{1}{x_2}.
\]

Continuing in the same way for \( c_2 \) and \( x_3 \) etc. we get

\[
x = c_0 + \frac{1}{c_1 + \frac{1}{c_2 + \frac{1}{c_3 + \ldots}}}
\]  

(18.37)

If \( x \) is a rational number (ratio of two integers) the continued fraction will eventually terminate. If \( x \) is irrational (like \( \pi \)) the continued fraction will go on for ever. The first few continued fraction coefficients \( c_i (i = 0, 1, 2 \cdots) \) for \( \pi \) are

\[
3, 7, 15, 1, \ldots
\]  

(18.38)

It is a property of continued fractions, which you can verify, that if a relatively large coefficient appears at some point, stopping the continued fraction at the previous coefficient gives an accurate approximation to the number. For, example, omitting 15 and subsequent coefficients in Eq. (18.38) gives the result

\[
3 + \frac{1}{7} = \frac{22}{7},
\]  

(18.39)

(where the continued fraction coefficients are in bold) which is well known to be a fairly good approximation for \( \pi \):

\[
\pi = 3.14159 \ldots, \quad \frac{22}{7} = 3.14286 \ldots
\]

(18.40)
In the present case we are interested in the continued fraction representation of $y/2^n$, which is a rational fraction so the continued fraction will eventually terminate. However, the value of $y/2^n$ is close to $m/r$ where $r$ is no bigger than $N$ ($N$ can be represented by $n_0$ qubits with $n_0 = n/2$). So we are interested in a continued fraction approximation to $y/2^n$ with a denominator no bigger than $N$. (Recall that $2^n = (2^{n_0})^2$ which is greater than $N^2$.)

Consider the example described in this chapter which has $N = 91, a = 4$ and $n = 14$ so $2^n = 16384$. The most probable results for $y$ are those in the column labeled “nearest integer” in Table 18.2. Suppose the measurement of $y$ gives the nearest integer for $m = 5$, i.e. 13653. The exact continued fraction coefficients of 13653/16384 are

$$0, 1, 4, 1, 1364, 2.$$  \hspace{1cm} (18.41)

Successive approximants are $0, 1, 4/5, 5/6, 6824/8189$ and 13653/16384. We want the approximant with the largest denominator less than $N$, i.e. 5/6, which tells us, if $m$ and $r$ have no common factors, that $m = 5$ and $r = 6$.

We check if $r = 6$ works by directly calculating $4^6( \text{ mod } 91)$. We find that it is equal to 1, see Eq. (18.29f), so the period is indeed 6. According to Appendix M in Mermin [Mer07] the probability of two large randomly chosen numbers not having a common factor is greater than 1/2. If we are unlucky and the assumption of no common factor does not work, then usually we would only have to try a few values for the common factor i.e. 2, 3, 4, · · ·, before succeeding. If we are really unlucky, and the common factor is very large, we would give up at some point, start again and get a different value for $y$. In the related example studied in detail in Sec. 18.6, where the measurement is assumed to give the nearest integer to the second peak, the common factor turns out to be 2.

### 18.C Unimportance of Small Phase Errors

The action of the controlled-phase gate is given by Eq. (18.11) and the QFT requires, in principle, these gates for $d = 1, 2, · · ·, n-1$. The total number of controlled phase gates is therefore $1+2+· · ·+n-1 = O(n^2)$. However, it is clearly impossible to accurately construct a phase gate for a phase which is exponentially small in $n$ if $n$ is large. For factoring, $n$ would typically be several thousand.

Fortunately it is not necessary to include controlled phase gates with such small phase changes. Mermin [Mer07] shows that one can generate the closest integer to a multiple of $2^n/r$ within almost the same probability as when one includes all gates (reduced by at most 1%) if one neglects controlled phase gates with $d > d^* = \log_2(Cn)$, where the constant $C$ is quite large (500π) but independent of $n$. Hence, in practice, one only needs of order $d^*n$ controlled phase gates ($\sim n \log_2 n$) to obtain the desired result, rather than $O(n^2)$ which would be needed if one includes all the gates with $d$ up to $n$. Hence the size of the circuit does not grow much faster than $n$ which is a huge improvement compared with $O(n^2)$ if $n$ is several thousand.
Chapter 19

Quantum Error Correction

19.1 Introduction

Quantum error correction has developed into a huge topic, so here we will only be able to describe the main ideas.

Error correction is essential for quantum computing, but appeared at first to be impossible, for reasons that we shall soon see. The field was transformed in 1995 by Shor [Sho95] and Steane [Ste96] who showed that quantum error correction is feasible. Before Shor and Steane, the goal of building a useful quantum computer seemed clearly unattainable. After those two papers, while building a quantum computer obviously posed enormous experimental challenges, it was not necessarily impossible.

Some general references on quantum error correction are Refs. [Mer07, NC00, Vat16, RP14].

Let us start by giving a simple discussion of classical error correction which will motivate our study of quantum error correction. Classically, error correction is not necessary for computation. This is because the hardware for one bit is huge on an atomic scale and the states 0 and 1 are so different that the probability of an unwanted flip is tiny. However, error correction is needed classically for transmitting a signal over large distances where it attenuates and can be corrupted by noise.

To perform error correction one needs redundancy. One simple way of doing classical error correction is to encode each logical bit by three physical bits, i.e.

\[
\begin{align}
|0\rangle & \rightarrow |\bar{0}\rangle \equiv |0\rangle|0\rangle|0\rangle \equiv |000\rangle, \\
|1\rangle & \rightarrow |\bar{1}\rangle \equiv |1\rangle|1\rangle|1\rangle \equiv |111\rangle,
\end{align}
\]

(for convenience we are using Dirac notation here even though these are classical bits for now.) The sets of three bits, |000\rangle and |111\rangle, are called codewords. One monitors the codewords to look for errors. If the bits in a codeword are not all the same one uses “majority rule” to correct. For example

|010\rangle is corrected to |000\rangle
|110\rangle is corrected to |111\rangle.

This works if no more than one bit is corrupted and so the error rate must be sufficiently low that the probability of two or more bits in a codeword being corrupted is negligible.

In quantum error correction one also uses multi-qubit codewords and monitoring. However, there are several major differences compared with classical error correction:

1. Error correction is essential. Quantum computing requires error correction. This is because the physical systems for a single qubit are very small, often on an atomic scale, so any small outside interference can disrupt the quantum state.
2. *Measurement destroys quantum information.* In contrast to the classical case checking for errors is problematic. Monitoring means measuring, and measuring a general quantum state alters it. Thus it seems that any attempt at error correction must destroy important quantum information.

3. *More general types of error can occur.* Bit flips are not the only possible errors. For example one can have phase errors where $\frac{1}{\sqrt{2}}(|0\rangle + |1\rangle) \rightarrow \frac{1}{\sqrt{2}}(|0\rangle + e^{i\phi}|1\rangle)$.

4. *Errors are continuous.* Unlike all-or-nothing bit flip errors for classical bits, errors in qubits can grow continuously out of the uncorrupted state.

One might imagine that point (ii), in particular, would be fatal. Amazingly this is not so as we shall see.

### 19.2 Correcting bit flip errors

We start our discussion of quantum error correction by considering how one might be able to correct just for bit flip errors. If the error rate is low we might hope to correct them by tripling the number of bits as in the classical case, Eq. (19.1).

The tripling of the qubits can be accomplished by the circuit in Fig. 19.1. To see how this works suppose that the input qubit, $|x\rangle$, is $|0\rangle$. Then none of the CNOT gates act on their target qubit so all three qubits are $|0\rangle$ at the end (i.e. on the right). However, if the input qubit $|x\rangle$ is $|1\rangle$ then the CNOT gates act so all three qubits are $|1\rangle$ at the end.

Figure 19.1: Circuit to encode the 3-qubit bit-flip code. Here $|x\rangle$ is $|0\rangle$ or $|1\rangle$ in the computational basis. The effect of this circuit on a linear combination of $|0\rangle$ and $|1\rangle$ is shown in Fig. 19.2.

\[
\alpha |0\rangle + \beta |1\rangle
\]

\[
|0\rangle \quad |x\rangle
\]

\[
|0\rangle \quad |0\rangle \quad |1\rangle
\]

\[
|0\rangle \quad |0\rangle \quad |0\rangle
\]

\[
|0\rangle \quad |0\rangle \quad |0\rangle
\]

\[
|0\rangle \quad |0\rangle \quad |0\rangle
\]

\[
|0\rangle \quad |0\rangle \quad |0\rangle
\]

\[
|0\rangle \quad |0\rangle \quad |0\rangle
\]

\[
|0\rangle \quad |0\rangle \quad |0\rangle
\]

\[
|0\rangle \quad |0\rangle \quad |0\rangle
\]

\[
|0\rangle \quad |0\rangle \quad |0\rangle
\]

\[
|0\rangle \quad |0\rangle \quad |0\rangle
\]

\[
|0\rangle \quad |0\rangle \quad |0\rangle
\]

\[
|0\rangle \quad |0\rangle \quad |0\rangle
\]

\[
|0\rangle \quad |0\rangle \quad |0\rangle
\]

\[
|0\rangle \quad |0\rangle \quad |0\rangle
\]

\[
|0\rangle \quad |0\rangle \quad |0\rangle
\]

\[
|0\rangle \quad |0\rangle \quad |0\rangle
\]

\[
|0\rangle \quad |0\rangle \quad |0\rangle
\]

\[
|0\rangle \quad |0\rangle \quad |0\rangle
\]

\[
|0\rangle \quad |0\rangle \quad |0\rangle
\]

\[
|0\rangle \quad |0\rangle \quad |0\rangle
\]

\[
|0\rangle \quad |0\rangle \quad |0\rangle
\]

\[
|0\rangle \quad |0\rangle \quad |0\rangle
\]

\[
|0\rangle \quad |0\rangle \quad |0\rangle
\]

\[
|0\rangle \quad |0\rangle \quad |0\rangle
\]

\[
|0\rangle \quad |0\rangle \quad |0\rangle
\]

\[
|0\rangle \quad |0\rangle \quad |0\rangle
\]

\[
|0\rangle \quad |0\rangle \quad |0\rangle
\]

\[
|0\rangle \quad |0\rangle \quad |0\rangle
\]

\[
|0\rangle \quad |0\rangle \quad |0\rangle
\]

\[
|0\rangle \quad |0\rangle \quad |0\rangle
\]

\[
|0\rangle \quad |0\rangle \quad |0\rangle
\]

\[
|0\rangle \quad |0\rangle \quad |0\rangle
\]

\[
|0\rangle \quad |0\rangle \quad |0\rangle
\]

\[
|0\rangle \quad |0\rangle \quad |0\rangle
\]

\[
|0\rangle \quad |0\rangle \quad |0\rangle
\]

\[
|0\rangle \quad |0\rangle \quad |0\rangle
\]

\[
|0\rangle \quad |0\rangle \quad |0\rangle
\]

\[
|0\rangle \quad |0\rangle \quad |0\rangle
\]

\[
|0\rangle \quad |0\rangle \quad |0\rangle
\]

\[
|0\rangle \quad |0\rangle \quad |0\rangle
\]

\[
|0\rangle \quad |0\rangle \quad |0\rangle
\]

\[
|0\rangle \quad |0\rangle \quad |0\rangle
\]

\[
|0\rangle \quad |0\rangle \quad |0\rangle
\]

\[
|0\rangle \quad |0\rangle \quad |0\rangle
\]

\[
|0\rangle \quad |0\rangle \quad |0\rangle
\]

\[
|0\rangle \quad |0\rangle \quad |0\rangle
\]

\[
|0\rangle \quad |0\rangle \quad |0\rangle
\]
By linearity a linear combination of $|0\rangle$ and $|1\rangle$ is transformed as we want:

$$\alpha|0\rangle + \beta|1\rangle \rightarrow \alpha|000\rangle + \beta|111\rangle, \quad (19.3)$$

see Fig. [19.2] Note that this is not a clone of the input state which would be

$$(\alpha|0\rangle + \beta|1\rangle)^\otimes 3 = \alpha^3|000\rangle + \alpha^2\beta(|001\rangle + |010\rangle + |100\rangle) + \alpha\beta^2(|110\rangle + |101\rangle + |011\rangle) + \beta^3|111\rangle. \quad (19.4)$$

We recall that cloning an arbitrary unknown state is impossible according to the no-cloning theorem.

Next an aside on notation. The CNOT gate is usually written with a $\oplus$ symbol (as in Figs. [19.1] and [19.2] to indicate the XOR operation but it is often more illuminating to write it, in an equivalent way, with an $X$ symbol (in a square) to indicate that the NOT (i.e. bit-flip) operation is performed with the operator $X$ (recall that in a CNOT gate the target qubit is flipped if the control qubit is 1.) From now on in this chapter we shall use the symbol $X$ inside a square when drawing a CNOT gate, see e.g. Fig. [19.3].

Now we have to check if any of the three qubits generated by the circuit in Fig. [19.2] are flipped so the situation is that shown in Fig. [19.3]. We assume that no more than one has been flipped, which is a reasonable approximation if the error rate is small.

Figure 19.3: Circuit indicating that at most one of the three bits generated by the circuit in Fig. [19.2] has flipped due to an error. The goal will be to determine whether any have flipped, if so which one, and then correct the error. Note that the Control-\textit{X} gates here are identical to the CNOT gates in Figs. [19.1] and [19.2] (Control-\textit{X} and CNOT are just different ways of describing the same gate.)

We have therefore one uncorrupted state and three corrupted states:

$$|\psi\rangle = \alpha|000\rangle + \beta|111\rangle, \quad (19.5a)$$

$$|\psi_1\rangle = \alpha|100\rangle + \beta|011\rangle \quad \text{(qubit 1 flipped)}, \quad (19.5b)$$

$$|\psi_2\rangle = \alpha|010\rangle + \beta|101\rangle \quad \text{(qubit 2 flipped)}, \quad (19.5c)$$

$$|\psi_3\rangle = \alpha|001\rangle + \beta|110\rangle \quad \text{(qubit 3 flipped)}. \quad (19.5d)$$

These four states are called the “syndromes”. Note that we denote the left hand qubit as the first qubit, the one to its right as the second qubit, and so on, e.g. $|x_1x_2x_3\rangle$. Hence in Eq. (19.5) $|\psi_i\rangle$ refers to the state in which qubit $i$ is flipped relative to the uncorrupted state $|\psi\rangle$.

Classically, to determine if one of the bits is flipped we just have to look at them. However, quantum mechanically, if we measure $|\psi\rangle$, say, we get $|000\rangle$ with probability $|\alpha|^2$ and $|111\rangle$ with probability $|\beta|^2$, which destroys the coherent superposition. It might therefore seem that quantum error correction is impossible.

Amazingly this is not so. The secret is to couple the codeword qubits to ancillary qubits and measure only these. This will give enough information to determine which syndrome the state is in without destroying the coherent superposition.
Here we need two ancillary qubits. The circuit including them is shown in Fig. 19.4. The three codeword qubits are at the bottom and the ancillary qubits are at the top. The ancillary qubits are measured and give values $x$ and $y$. We shall now see that each of the four possible pairs of values for $x$ and $y$ corresponds to one of the syndrome states in Eq. (19.5).

Both ancillas are targeted by two of the codeword qubits.

1st (upper) ancilla ($x$) is targeted by codeword qubits 1 and 2.
2nd (lower) ancilla ($y$) is targeted by codeword qubits 2 and 3.

Let’s see what happens for the four syndrome states.

$|\psi\rangle$ Codeword $|000\rangle$. No ancilla flipped so $x = 0, y = 0$.
Codeword $|111\rangle$. Both ancillas are flipped twice so again $x = 0, y = 0$.
Note that the result of the measurement is the same for both the $|000\rangle$ and $|111\rangle$ parts of the state $|\psi\rangle$. Hence the coherent superposition of $|\psi\rangle$ is not destroyed by the measurement on the ancillas.

$|\psi_1\rangle$ Codeword $|100\rangle$. $x$ is flipped once, and $y$ is not flipped, so $x = 1, y = 0$.
Codeword $|011\rangle$. $x$ is flipped once and $y$ is flipped twice so again $x = 1, y = 0$.
Recall that the qubits are ordered such that qubit 1 is on the left.

$|\psi_2\rangle$ Codeword $|010\rangle$.
Codeword $|101\rangle$. $x$ and $y$ are both flipped once so $x = 1, y = 1$.
Codeword $|001\rangle$. $x$ is not flipped and $y$ is flipped once so $x = 0, y = 1$.
Codeword $|110\rangle$. $x$ is flipped twice and $y$ is flipped once so again $x = 0, y = 1$.

Hence we get the table of results shown in Table 19.1. Note that in all cases the coherent superposition of the syndrome state is not destroyed by the measurement of the ancillas.

Hence by measuring the auxiliary qubits we can determine which if any of the codeword qubits have flipped and then apply a compensating flip if necessary. The $X$-gates which perform these compensating flips are shown at the right of Fig. 19.4. For example the $X^{\tilde{y}}$ gate on qubit 1 indicates that a flip is done by acting with the $X$ operator on qubit 1 only if $x = 1$ and $y = 0$, which corresponds to the second entry in the Table 19.1 ($\tilde{y}$ means the complement of $y$).
19.2. CORRECTING BIT FLIP ERRORS

We have assumed up to now that the state of the system has had a bit flipped with probability one. However, as already noted, errors in quantum circuits can arise continuously from zero, and we are concerned with the situation in which the error rate is small (otherwise we cannot error correct). Consider therefore a state $|\psi\rangle$ which has a small amplitude less than one to have a bit flipped, i.e.

$$|\psi\rangle \rightarrow [1 + i(\epsilon_1 X_1 + \epsilon_2 X_2 + \epsilon_3 X_3)]|\psi\rangle,$$

(19.6)

where $\epsilon_k \ll 1$, and we have only indicated terms to first order in the $\epsilon_k$ (the factor of $i$ is needed so the state is normalized to first order in the $\epsilon_k$). The probability of qubit $k$ being flipped is $|\epsilon_k|^2$ to leading order. When we measure the ancilla qubits we project on to either the uncorrupted state or one of the three corrupted states which have one qubit flipped.

Since the $\epsilon_k$ are small, the probability that a corrupted state is detected is small, so the most probable situation is that no correction is needed. However, there is a small probability that the projection will be on to one of the corrupted syndromes. The corrupted syndromes differ substantially from the uncorrupted state. They are further, in fact, from the uncorrupted state than the original state in Eq. (19.6). This might, at first, seem like a retrograde step but it is not because the corrupted state is known precisely so it is possible to correct it back to the uncorrupted state.

To summarize this part, quantum error correction is feasible, even though errors arise continuously, because possibly corrupted states are projected on to one of a discrete set of states which can be corrected if necessary. We will discuss this important point again in Sec. 19.5 when we consider how general errors arise.

It should be noted that in classical analog computers, where errors also arise continuously, no such projection can be done, and hence error correction cannot be performed. This is why we don’t have classical analog computers.

Going back to the discussion of Fig. 19.4 one can avoid explicitly measuring the qubits and instead coherently and automatically correct any bit-flip error by having the ancillas interact back on the codeword qubits as shown in Fig. 19.5. In that figure, the rightmost three controlled gates have the same effect as the NOT gates in the right of Fig. 19.3 which depend on the result of measurements of the $x$ and $y$ ancillary qubits. The rightmost gate in Fig. 19.5 has two control qubits and three target qubits. This gate flips all the target qubits if both control qubits are 1. It is a generalization of the Toffoli gate $T$ which has two control qubits, and one target qubit which is flipped if both control qubits are 1, i.e. $T|x\rangle|y\rangle|z\rangle = |x\rangle|y\rangle|z \oplus x y\rangle$. If we denote by $T^*$ the rightmost gate in Fig. 19.5 then $T^*|x\rangle|y\rangle|z\rangle|u\rangle|v\rangle = |x\rangle|y\rangle|z \oplus x y\rangle|u \oplus x y\rangle|v \oplus x y\rangle$. Note that this gate is equivalent to three separate Toffoli gates, in which the two ancilla qubits are the controls, qubit 1 is the target for the first Toffoli, qubit 2 for the second Toffoli, etc. In Vathsan’s book [Vat16], her Fig. 10.5 shows a set of gates which is equivalent to, but different from, the rightmost three in Fig. 19.5, including a zero-controlled CNOT gate (indicated by the open circle in her Fig. 10.5). After the error on the computational bits has been corrected the ancilla qubits have to be reinitialized to zero.

It is instructive to show for the different syndromes in Eq. (19.5) that the circuits in Figs. 19.4 and 19.5 give the same result, i.e. the end product is the uncorrupted state $|\psi\rangle$. The results from the circuit

<table>
<thead>
<tr>
<th>syndrome</th>
<th>bit flipped</th>
<th>x</th>
<th>y</th>
</tr>
</thead>
<tbody>
<tr>
<td>$</td>
<td>\psi\rangle$</td>
<td>none</td>
<td>0</td>
</tr>
<tr>
<td>$</td>
<td>\psi_1\rangle$</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>$</td>
<td>\psi_2\rangle$</td>
<td>2</td>
<td>1</td>
</tr>
<tr>
<td>$</td>
<td>\psi_3\rangle$</td>
<td>3</td>
<td>0</td>
</tr>
</tbody>
</table>

Table 19.1: Results of measurement of the ancillary qubits for the different syndromes of the codeword qubits.
Figure 19.5: Automation of the error correction procedure of Fig. 19.4. The three controlled gates on the right have the same effect as the NOT gates on the right of Fig. 19.4 which depend on the result of measurements of the $x$ and $y$ ancillary qubits. The rightmost gate, with two control qubits and three target qubits, is discussed in the text. The values of the control bits $x$ and $y$ at the end depend on which of the four syndromes is present (i.e. which if any of the $X$ gates on the left of the figure have acted) according to Table 19.1.

of Fig. 19.4 have already been discussed above. For the circuit in Fig. 19.5 we just consider the case of $|\psi_2\rangle$ (so qubit 2 has been flipped), and we have $x = 1, y = 1$ according to Table 19.1. Consider the rightmost three gates in Fig. 19.5 (these are the ones that do the error correction). For $x = 1, y = 1$, the rightmost gate is active and flips all three codeword qubits. Hence, between them, the rightmost three gates flip codeword qubit 1 twice, flip codeword qubit 2 once, and flip codeword qubit 3 twice. The net result is that only codeword qubit 2 is flipped so we recover the uncorrupted state $|\psi\rangle$. It is useful to check that the circuit in Fig. 19.5 also works to correct $|\psi_1\rangle$ and $|\psi_3\rangle$.

19.3 Stabilizer formalism

In order to conveniently generalize the ideas in the previous section to arbitrary errors we need to reformulate them.

For reasons that will shortly become clear, consider the two Hermitian operators $Z_1Z_2$ and $Z_2Z_3$. Because $Z_i^2 = 1$ (the identity) and different $Z$’s commute we have

$$ (Z_1Z_2)^2 = 1, \quad (Z_2Z_3)^2 = 1. \quad (19.7) $$

An operator whose square is unity has eigenvalues equal to ±1, since acting twice with the operator on an eigenvector gives the eigenvector, so the square of the eigenvalue is 1. We also know that $Z_1Z_2$ and $Z_2Z_3$ commute with each other.

One can verify that the syndrome states in Eq. (19.5) are eigenvectors of $Z_1Z_2$ and $Z_2Z_3$ according to Table 19.2. The connection between the values of the stabilizers, $Z_1Z_2$ and $Z_2Z_3$, and the measured values of the ancilla qubits, $x$ and $y$, is $Z_1Z_2 = (-1)^x$, $Z_2Z_3 = (-1)^y$. In general we use the term “stabilizers” to denote operators like operators $Z_1Z_2$ and $Z_2Z_3$ whose eigenvalues distinguish the different syndromes.

As discussed in Chapter 3 it is an axiom of quantum mechanics that measurable quantities are represented by Hermitian operators.
There is a more straightforward way to determine whether the eigenvalue of a stabilizer in a syndrome is $+1$ or $-1$ than acting with the stabilizer on the syndrome.

We note first that the eigenvalue of all the stabilizers is $+1$ in the uncorrupted syndrome $|\psi\rangle$. This is an important property that stabilizers must have.

Also note that the operators for the stabilizers will be built out of the single-qubit operators $X, Y$. For the 3-qubit, bit-flip code we only have the $Z$ but the $X$ will also be needed to correct for general errors. Furthermore the syndromes with a single qubit error are obtained by acting on the uncorrupted syndrome with the $X_i, Y_i$ and $Z_i$ operators.

Again, for our simple example above, we only had the $X_i$, but the other operators will be used when we deal with general errors.

The operators, $X_i, Y_i, Z_i$, have the property that they commute for different qubits $i$, and anti-commute for the same qubit, where the anti-commutator of $A$ and $B$ is defined by $\{A, B\} \equiv AB + BA$. Hence we have, for example,

$$[X_i, Y_j] \equiv X_i Y_j - Y_j X_i = 0 \quad (i \neq j),$$

$$\{X_i, Y_j\} \equiv X_i Y_j + Y_j X_i = 0. \quad (19.8a)$$

(Verify the anti-commutation relations like Eq. (19.8b) by explicitly working out some cases.)

Consequently, if we consider a general stabilizer $A_\alpha$ and a syndrome state $|\psi_\beta\rangle = B_\beta |\psi\rangle$ then $A_\alpha$ either commutes or anti-commutes with $B_\beta$. We will now show that if $A_\alpha$ commutes with $B_\beta$ the eigenvalue of the stabilizer $A_\alpha$ in state $|\psi_\beta\rangle$ is $+1$ and if they anti-commute the eigenvalue is $-1$.

Firstly, if $A_\alpha$ commutes with $B_\beta$ then

$$A_\alpha |\psi_\beta\rangle = A_\alpha B_\beta |\psi\rangle = B_\beta A_\alpha |\psi\rangle = B_\beta |\psi\rangle = |\psi_\beta\rangle, \quad (19.9)$$

where we used that the eigenvalues of all the stabilizers $A_\alpha$ are $+1$ in the uncorrupted state $|\psi\rangle$ to get the second equality. Hence the eigenvalue of $A_\alpha$ in state $|\psi_\beta\rangle$ is $+1$ if $A_\alpha$ commutes with $B_\beta$. Similarly if $A_\alpha$ anti-commutes with $B_\beta$ then

$$A_\alpha |\psi_\beta\rangle = A_\alpha B_\beta |\psi\rangle = -B_\beta A_\alpha |\psi\rangle = -B_\beta |\psi\rangle = -|\psi_\beta\rangle, \quad (19.10)$$

so the eigenvalue is $-1$.

Next we will see how to determine if a stabilizer commutes or anti-commutes with the operator which generates a corrupted syndrome out of the uncorrupted state.

---

2Recall that the Pauli operators $X, Y$ and $Z$ are given by $X \equiv \sigma^x = \begin{pmatrix} 0 & 1 \\ 1 & 0 \end{pmatrix}, Y \equiv \sigma^y = \begin{pmatrix} 0 & -i \\ i & 0 \end{pmatrix}, Z \equiv \sigma^z = \begin{pmatrix} 1 & 0 \\ 0 & -1 \end{pmatrix}$ and so $Y = i X Z$.
For the case of the 3-qubit, bit-flip code discussed so far the stabilizers are
\[ Z_1Z_2 \text{ and } Z_2Z_3, \] (19.11)
and the operators which generate the corrupted syndrome from the uncorrupted state are
\[ X_1, X_2 \text{ and } X_3. \] (19.12)
As an example, we see that \( X_1 \) commutes with \( Z_2Z_3 \) because there are no sites in common, so the eigenvalue of \( Z_2Z_3 \) for \( |\psi_1\rangle \) must be +1 which agrees with Table [19.2]. On the other hand \( X_2 \) has one site in common with \( Z_2Z_3 \) so
\[ X_2Z_2Z_3 = -Z_2X_2Z_3 = -Z_2Z_3X_2, \] (19.13)
and the operators anticommute, so the eigenvalue of \( Z_2Z_3 \) for \( |\psi_2\rangle \) must be -1, which again agrees with Table [19.2]. Every time we have to interchange the order of two different operators acting on the same qubit we pick up a minus sign. Hence it is straightforward to deduce the overall sign. Note that operators of the same type, e.g. the \( Z_i \), always commute.

As a more complicated example, which occurs in a scheme for full error correction, consider the stabilizer \( Z_3X_4X_5Z_1 \). For the syndrome which has been corrupted by \( Z_4 \) the eigenvalue is -1, the minus sign coming from interchanging the order of \( X_4 \) and \( Z_4 \). However, for the syndrome which was corrupted by \( X_4 \) the eigenvalue is +1 since, for the qubit in common, i.e. 4, both operators are \( X \) and so commute. As another example, for the syndrome which was corrupted by \( X_2 \) the eigenvalue is +1, because \( X_2 \) and the stabilizer commute since they have no qubits in common.

To summarize, in the stabilizer formalism we need to construct a mutually commuting set of Hermitian operators (the stabilizers) which square to 1 and for which (i) the syndromes are eigenstates, (ii) the uncorrupted syndrome has eigenvalue +1 for all stabilizers, and (iii) the set of \( \pm 1 \) eigenvalues of the stabilizers uniquely specifies the syndrome. In Sec. [19.6] we will describe an example with full error correction which has codewords with 9 qubits and needs 8 stabilizers.

Figure 19.6: A circuit with a control-\( U \) gate in which the control (upper) qubit is surrounded by Hadamards. \( U \) is an operator with eigenvalues \( \pm 1 \) and corresponding eigenvectors \( |\psi_+\rangle \) and \( |\psi_-\rangle \). As shown in the text, if a measurement of the upper qubit gives \( |0\rangle \) then the lower qubit will be in state \( |\psi_+\rangle \), and if the measurement gives \( |1\rangle \) then the lower qubit will be in state \( |\psi_-\rangle \). The states \( |\phi_i\rangle \ (i = 0, 1, 2, 3) \) are described in the text. Note that this figure is identical to Fig. 8.8 and was discussed in Chapter 8.

In terms of stabilizers, what circuit which will determine which syndrome has occurred? Consider the circuit in Fig. 19.6 which includes a control-\( U \) gate where \( U \) is an operator, which, like the stabilizers, has eigenvalues \( \pm 1 \). If the control qubit is 1 the effect on the target qubit is
\[ U|\psi_+\rangle = |\psi_+\rangle, \quad U|\psi_-\rangle = -|\psi_-\rangle, \] (19.14)
where $|\psi_+\rangle$ and $|\psi_-\rangle$ are the eigenvectors with eigenvalue $+1$ and $-1$ respectively. If the control qubit is 0 then the target qubit is unchanged. We discussed this circuit in Chapter 8 and found that the states $|\phi_i\rangle$, $(i = 0, 1, 2, 3)$ are given by Eqs. (8.19). In particular, the final state $|\phi_3\rangle$, before the measurement of the upper qubit, is given by

$$|\phi_3\rangle = \alpha_+|0\psi_+\rangle + \alpha_-|1\psi_-\rangle.$$

(19.15)

Hence if a measurement of the upper qubit gives $|0\rangle$ (which it does with probability $|\alpha_+|^2$) the lower qubit will be in state $|\psi_+\rangle$, and if the measurement gives $|1\rangle$ (probability is $|\alpha_-|^2$) the lower qubit will be in state $|\psi_-\rangle$. We see that measuring the control qubit tells us which eigenstate of $U$ the target qubit is in.

Stabilizers involve more than one codeword qubit so the gates we need will have several target qubits. For the 3-qubit, bit-flip code, the circuit equivalent to Fig. 19.4 is shown in Fig. 19.7.

Figure 19.7: Circuit equivalent to that in Fig. 19.4 but in the stabilizer formalism. In this circuit $x$ measures $Z_1Z_2$, and $y$ measures $Z_2Z_3$. In other words, if $x = 0$ the state of the codeword bits has $Z_1Z_2 = +1$, whereas if $x = 1$ the state of the codeword bits has $Z_1Z_2 = -1$, with an analogous correspondence between $y$ and $Z_2Z_3$. Note that $Z_1Z_2$ and $Z_2Z_3$ have eigenvalues $\pm 1$ and commute with each other.

The equivalence of the circuits in Figs. 19.4 and 19.7 can also be understood from the simpler case of the equivalences shown in Fig. 19.8 in which the left-hand equality comes from the fact that the target and control qubits can be exchanged in a control-Z gate\(^3\) and the right-hand equality is because $HZH = X$ and $H^2 = 1$ (the identity).

\(^3\)Because the only effect of the gate is to change the sign of the state if both target and control qubits are 1.

Figure 19.8: The equalities in this figure are helpful to understand the equivalence of Figs. 19.4 and 19.7. The left-hand equality comes from the fact that the target and control qubits can be exchanged in a control-Z gate, and the right-hand equality is because $HZH = X$ and $H^2 = 1$.

The stabilizer formalism will be convenient when devising circuits for full error correction rather than just correcting bit flips as we have done up to now.
CHAPTER 19. QUANTUM ERROR CORRECTION

19.4 Phase Flip Code

Before discussing how to correct general errors, we will briefly mention another special case, a phase flip, which has no classical equivalent since classical bits don’t have any property corresponding to phase. In this error model, with some probability $p$, the relative phase of $|0\rangle$ and $|1\rangle$ is flipped so

$$ |\psi\rangle = \alpha|0\rangle + \beta|1\rangle \rightarrow \alpha|0\rangle - \beta|1\rangle. $$

Equivalently

$$ \begin{pmatrix} \alpha \\ \beta \end{pmatrix} \rightarrow Z \begin{pmatrix} \alpha \\ \beta \end{pmatrix} = \begin{pmatrix} \alpha \\ -\beta \end{pmatrix} \quad \text{(computational basis).} $$

The phase-flip error model can be turned into the already-studied bit-flip model by transforming to the $\pm$ basis (also called the $X$-basis because it is the basis in which $X$ is diagonal) where

$$ + = \frac{1}{\sqrt{2}} (|0\rangle + |1\rangle), \quad - = \frac{1}{\sqrt{2}} (|0\rangle - |1\rangle), $$

One transforms between the $\pm$ basis and the computational basis using Hadamards:

$$ H|0\rangle = |+\rangle, \quad H|1\rangle = |-\rangle, \quad H|+\rangle = |0\rangle, \quad H|-\rangle = |1\rangle. $$

In the $\pm$ basis the roles of $X$ and $Z$ are interchanged since

$$ X|0\rangle = |1\rangle, \quad X|1\rangle = |0\rangle, \quad Z|0\rangle = |0\rangle, \quad Z|1\rangle = -|1\rangle, \quad Z|+\rangle = |-\rangle, \quad Z|-\rangle = |+\rangle, \quad X|+\rangle = |+\rangle, \quad X|-\rangle = |-\rangle. $$

Thus we shall find in Sec. 19.6 that stabilizers to detect phase errors involve $X$ operators, as opposed to those used to detect bit-flip errors which involve $Z$ operators (see Fig. 19.7).

$$ \begin{array}{c}
\alpha|0\rangle + \beta|1\rangle \\
|0\rangle \quad X \\
|0\rangle \\
\end{array} \quad H \\
\quad H \\
\quad \alpha|+\rangle + \beta|-\rangle$$

Figure 19.9: Encoding circuit for the 3-qubit phase-flip code.

The encoding circuit for the 3-qubit phase-flip code is obtained from that for the 3-qubit bit-flip code in Fig. 19.2 by adding Hadamards to the circuit, with the result shown in Fig. 19.9. We shall use this circuit in Sec. 19.6 as part of the encoding circuit in Fig. 19.10 for a code (due to Shor) which corrects general 1-qubit errors.

19.5 The Physics of Error Correction (discretization of errors)

In our discussion of errors we have so far implicitly assumed that the errors occur because of some malfunction in the circuit. The state has undergone a unitary transformation, but not exactly the
right one. Another, and very important, source of error is interaction between the qubits and the environment, which is unavoidable even though quantum computer engineers work very hard to reduce it to a minimum. This can lead to errors due to a non-unitary change in the computational qubits (though the combined system of qubits plus environment undergoes unitary time development.) In this section we include the effects of the environment and also consider the most general type of single qubit error.

Consider a single qubit $|x\rangle$, and call the environment $|e\rangle$. Unlike the state of the qubit, the state of the environment is in a space of very many dimensions. Ideally $|x\rangle$ evolves under the effects of the gates only, independent of the environment. However, interactions with the environment cannot be avoided which leads to a corruption of the qubit and an entangling of the qubit with the environment.

The most general such form of these effects is

$$|e\rangle |0\rangle \rightarrow |e_0\rangle |0\rangle + |e_1\rangle |1\rangle,$$

$$|e\rangle |1\rangle \rightarrow |e_2\rangle |0\rangle + |e_3\rangle |1\rangle,$$  

where $|e_i\rangle$ ($i = 0,\cdots, 3$) are possible final states of the environment. The environment states are not normalized, and not orthogonal either. However, the two states on the right hand side of Eqs. (19.21) must be orthogonal since the time evolution of the combined qubit-environment system is unitary. In other words

$$\langle e_2 | e_0 \rangle + \langle e_3 | e_1 \rangle = 0.$$  

(19.22)

The corruption of the computation by the environment indicated in Eq. (19.21) is called “decoherence”. It is the main source of difficulty in building a practical quantum computer.

In previous sections we have neglected entanglement with the environment. Rather, errors were assumed to occur because of mistakes made in the circuit itself. This corresponds to a special case of Eqs. (19.21), where all the environment states are the same, apart from normalization, i.e. $|e_i\rangle = c_i|\rangle$, for $i = 0,\cdots, 3$.

We are interested in the case where the probability of an error is small (otherwise we would not be able to correct for it), i.e.

$$\langle e | e \rangle = 1, \quad \langle e_0 | e_0 \rangle \simeq 1, \quad \langle e_3 | e_3 \rangle \simeq 1, \quad \langle e_1 | e_1 \rangle \ll 1, \quad \langle e_2 | e_2 \rangle \ll 1.$$  

(19.23)

Equations (19.21) can be combined into one as

$$|e\rangle |x\rangle \rightarrow \left\{ \left( \frac{|e_0\rangle + |e_3\rangle}{2} \right) 1 + \left( \frac{|e_0\rangle - |e_3\rangle}{2} \right) Z + \left( \frac{|e_2\rangle + |e_1\rangle}{2} \right) X + \left( \frac{|e_2\rangle - |e_1\rangle}{2} \right) (iY) \right\} |x\rangle,$$  

(19.24)

where $x = 0$ or 1 and, as usual\(^4\)

$$Z = \begin{pmatrix} 1 & 0 \\ 0 & -1 \end{pmatrix}, \quad X = \begin{pmatrix} 0 & 1 \\ 1 & 0 \end{pmatrix}, \quad iY = ZX = \begin{pmatrix} 0 & 1 \\ -1 & 0 \end{pmatrix}, \quad 1 = \begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix}.$$  

(19.25)

Please evaluate Eq. (19.25) separately for $x = 0$ and 1 to verify that it is equivalent to Eqs. (19.21). There is nothing special about these environment states so we can write

$$|e\rangle |x\rangle \rightarrow (|d\rangle 1 + |a\rangle X + |b\rangle (iY) + |c\rangle Z ) |x\rangle.$$  

(19.26)

Equation (19.25) applies to both $x = 0$ and $x = 1$. Since time evolution of the combined qubit-environment system follows quantum mechanics and so is unitary and linear, it also applies to a linear superposition $|\psi\rangle = \alpha |0\rangle + \beta |1\rangle$ so

$$|e\rangle |\psi\rangle \rightarrow (|d\rangle 1 + |a\rangle X + |b\rangle (iY) + |c\rangle Z ) |\psi\rangle.$$  

(19.27)

\(^4\)I prefer to write equations like (19.21) in terms of $iY (= ZX)$ rather than $Y$ to avoid having explicitly complex coefficients. Many texts on quantum computing write $ZX$ rather than $iY$. Note that $iY (= ZX)$ is not Hermitian (though $Y$ is) but we do not need the Hermitian property here. More importantly, $iY$, like $X,Y$ and $Z$ is unitary.
We see that the effects of the environment on the uncorrupted state of a single qubit can be expressed entirely in terms of the Pauli operators, $X$, $(iY)$ and $Z$. These are characterized as follows:

- $X$ corresponds to a bit-flip error,
- $Z$ corresponds to a phase-flip error, and
- $iY (= ZX)$ corresponds to a combined bit-flip and phase-flip error.

Intuitively, the reason that the new state can be expressed in terms of the Pauli operators and the identity, is that any $2 \times 2$ matrix can be written as a linear combination of these operators, see Eq. (2.21).

We remind the reader that the environment states are not normalized, and so, in the important case where the initial state is close to the final state, we have

$$
\langle a | a \rangle \ll 1, \quad \langle b | b \rangle \ll 1, \quad \langle c | c \rangle \ll 1,
$$

(19.28)

in Eq. (19.27),

We now extend this discussion to the situation where we have expanded a single qubit into an $n$-qubit codeword which we write as $| \psi \rangle_n$. In this course we just consider how to correct single-qubit errors, so we neglect the possibility that two or more of the qubits in the codeword are corrupted. From Eq. (19.27), we see that all single qubit errors are incorporated by

$$
| e \rangle | \psi \rangle_n \rightarrow \left( | d \rangle \right. 1 + \sum_{k=1}^{n} | a_k \rangle X_k + \sum_{k=1}^{n} | b_k \rangle (iY)_k + \sum_{k=1}^{n} | c_k \rangle Z_k \left. \right) | \psi \rangle_n.
$$

(19.29)

Based on Eq. (19.29), single qubit quantum error correction involves the following steps:

- Expand the logical qubit to an $n$-qubit codeword.
- Project the possibly corrupted state to one of the $3^n + 1$ states (syndromes) on the right hand side of Eq. (19.29), with information indicating which one.
- Correct, if necessary, the 1-qubit error by acting with the appropriate $X_k, Y_k$ or $Z_k$.

Note:

1. The whole continuum of errors can be represented by a finite set of discrete errors. Errors emerge continuously from the uncorrupted state by increasing from zero the size of the terms in Eq. (19.29) involving $X_i, Y_i$ and $Z_i$, which are characterized by $\langle a_i | a_i \rangle^{1/2}, \langle b_i | b_i \rangle^{1/2}$ and $\langle c_i | c_i \rangle^{1/2}$ respectively. However, the projection is always to one of the $3n+1$ discrete states. If the amplitude of the error is small then, with high probability, the projection will be to the uncorrupted state (which needs no correction) but with small but non-zero probability the projection will be to one of the $3n$ corrupted states (which do need correction).

2. An arbitrary error on a single qubit will be corrected, not just bit-flip ($X$), or phase-flip ($Z$), or combined bit- and phase-flip ($iY$) errors but also any combination of them. For example, suppose that the $k$-th qubit has been reinitialized to zero, i.e.

$$
| 0_k \rangle \rightarrow | 0_k \rangle, \quad | 1_k \rangle \rightarrow | 0_k \rangle.
$$

(19.30)
The matrix which accomplishes this transformation is
\[
\begin{pmatrix}
1 & 1 \\
0 & 0
\end{pmatrix}
\] (19.31)
which can be written as
\[
\frac{1 + X_k + iY_k + Z_k}{2}.
\] (19.32)
Hence the state of the codeword qubits and environment has been transformed as follows:
\[
|e\rangle |\psi\rangle_n \rightarrow |e'\rangle |\psi'\rangle_n = |e'\rangle \frac{1}{2} (1 + X_k + iY_k + Z_k) |\psi\rangle_n.
\] (19.33)
The codeword qubits are now in a linear combination of four syndromes, corresponding to the four terms in this equation. A general syndrome measuring circuit, such as the Shor 9-qubit code discussed in the next section, will detect these syndromes and obtain a unique set of values for the ancilla qubits for each of them. Hence, even for this non-unitary error, measuring the ancillas will project on to one of the syndromes which can then be corrected if necessary.

3. A full discussion of how the entanglement of qubits with the environment generates errors and how they can subsequently be corrected, requires the formalism of the density matrix, see Chapter 5 and Refs. [NC00, RP14]. This advanced treatment of quantum error correction is discussed in Refs. [NC00, RP14] but is beyond the scope of the course.

19.6 Correcting arbitrary Errors: the Shor Code

In the section we discuss a code, due to Peter Shor [Sho95], for correcting arbitrary 1-qubit errors. This code needs code words of nine qubits to represent one logical qubit. It is not the most efficient code, there are others which use smaller code words and so don’t need as many physical qubits, but the structure of Shor’s code follows quite naturally from the discussion we have already given of 1-qubit bit-flip, and 1-qubit phase-flip errors, so will discuss it here.

Essentially it combines bit-flip (X) and phase-flip (Z) codes, which turns out to then automatically correct for combined bit-flip, phase-flip (iY) errors. As discussed in the previous section, it then also corrects arbitrary 1-qubit errors.

We first encode for phase flips:
\[
|0\rangle \rightarrow |++\rangle, \quad |1\rangle \rightarrow |--\rangle,
\] (19.34)
and then encode for bit-flip errors
\[
|+\rangle \rightarrow \frac{1}{\sqrt{2}} (|000\rangle + |111\rangle), \quad |--\rangle \rightarrow \frac{1}{\sqrt{2}} (|000\rangle - |111\rangle).
\] (19.35)
The final result is the 9-qubit encoding
\[
|0\rangle \rightarrow |\bar{0}\rangle = \frac{1}{2^{3/2}} (|000\rangle + |111\rangle) (|000\rangle + |111\rangle) (|000\rangle + |111\rangle), \quad (19.36a)
|1\rangle \rightarrow |\bar{1}\rangle = \frac{1}{2^{3/2}} (|000\rangle - |111\rangle) (|000\rangle - |111\rangle) (|000\rangle - |111\rangle). \quad (19.36b)
\]

The reader will notice that the transformation in Eqs. (19.31), which involves a linear combination of X, iY and Z on a single qubit, are not unitary. Now the evolution of an isolated (closed) system is unitary, However, qubits are coupled to the environment. If we consider a system coupled to the environment (called an open system), and subject the combined system+environment to a unitary transformation, and finally consider the behavior of just the system by tracing out over the environment, the resulting transformation of the system is not necessarily unitary [NC00, RP14].
These two equations can be combined as
\[
|x\rangle \rightarrow |\overline{x}\rangle = \frac{1}{2^{3/2}} \left( |000\rangle + (-1)^x |111\rangle \right) \left( |000\rangle + (-1)^x |111\rangle \right) \left( |000\rangle + (-1)^x |111\rangle \right), \tag{19.37}
\]
or more concisely as
\[
|\overline{x}\rangle = \frac{1}{2^{3/2}} \left( |000\rangle + (-1)^x |111\rangle \right)^{\otimes 3}. \tag{19.38}
\]
Such a code is called a concatenated code. The circuit to achieve this encoding is obtained by concatenating the phase flip and the bit flip encodings as shown in Fig. 19.10. Note the labeling of the qubits. The qubits in each of the three blocks in Eq. (19.36) have labels 123, 456, and 789.

Figure 19.10: Encoding for the Shor 9-qubit code. The initial state at the top left, $|x\rangle$, is equal to $|0\rangle$ or $|1\rangle$ in the computational basis, so $\phi_0 = |xxx\rangle$ and $\phi_1 = 2^{-3/2}(|0\rangle + (-1)^x |1\rangle) (|0\rangle + (-1)^x |1\rangle) (|0\rangle + (-1)^x |1\rangle)$ since $H|x\rangle = 2^{-1/2}(|0\rangle + (-1)^x |1\rangle)$. We see by comparison with Fig. 19.1 that if $x = 0$ then the final state $|\phi_2\rangle$ is equal to $|0\rangle$ given by Eq. (19.36a), while if $x = 1$ the final state is $|1\rangle$ given by Eq. (19.36b). If the initial state at the top left is a linear combination $\alpha |0\rangle + \beta |1\rangle$ then, by linearity, the final state at the right is $\alpha |0\rangle + \beta |1\rangle$. The numbers at the right are the labels of the nine qubits. Note that this circuit is a concatenation of the encoding circuit for phase-flips shown in Fig. 19.9 and that for bit-flips in Fig. 19.1.

The form of the 1-qubit corruption in Eq. (19.29) simplifies a little here because if $|\psi\rangle$ is a linear combination of the codeword states in Eq. (19.36) then
\[
Z_1 |\psi\rangle = Z_2 |\psi\rangle = Z_3 |\psi\rangle, \tag{19.39a}
Z_4 |\psi\rangle = Z_5 |\psi\rangle = Z_6 |\psi\rangle, \tag{19.39b}
Z_7 |\psi\rangle = Z_8 |\psi\rangle = Z_9 |\psi\rangle, \tag{19.39c}
\]
since a $Z_i$ operator changes one of the + signs in Eq. (19.36a) into a $-$ sign, and the corresponding $-$ sign in Eq. (19.36b) into a + sign.
19.6. CORRECTING ARBITRARY ERRORS: THE SHOR CODE

Figure 19.11: A circuit to measure the error syndrome for the Shor 9-qubit code. The nine codeword qubits are at the bottom and the eight ancillary qubits at the top. The ancillary qubits determine the values of the eight, mutually commuting stabilizers in Eq. (19.41), $M_1 = Z_1Z_2$, $M_2 = Z_2Z_3$, $M_3 = Z_4Z_5$, $M_4 = Z_5Z_6$, $M_5 = Z_7Z_8$, $M_6 = Z_8Z_9$, $M_7 = X_1X_2X_3X_4X_5X_6$, and $M_8 = X_4X_5X_6X_7X_8X_9$.

The nine codeword qubits can be conveniently grouped into three blocks of three, containing qubits $123$, $456$ and $789$ respectively. $M_1$ and $M_2$ act entirely on the first block, and do so in the same way as

$$\langle e|\psi\rangle \rightarrow \left| d \right|I + \left| c \right|Z_1 + \left| c' \right|Z_4 + \left| c'' \right|Z_7 + \sum_{i=1}^{9} \left| a_i \right|X_i + \sum_{i=1}^{9} \left| b_i \right|iY_i \right| \psi\rangle. \quad (19.40)$$

The eight stabilizers which we use to diagnose the error are

$$M_1 = Z_1Z_2, \quad M_2 = Z_2Z_3, \quad M_3 = Z_4Z_5, \quad M_4 = Z_5Z_6, \quad M_5 = Z_7Z_8, \quad M_6 = Z_8Z_9, \quad M_7 = X_1X_2X_3X_4X_5X_6, \quad M_8 = X_4X_5X_6X_7X_8X_9. \quad (19.41)$$

Note that the nine qubits can conveniently be grouped into three blocks of three, containing qubits 123, 456 and 789 respectively. $M_1$ and $M_2$ act entirely on the first block, and do so in the same way as...
the stabilizers of the 3-qubit, bit flip code shown in Fig. [19.7] Similarly $M_3$ and $M_4$ act on the second block and $M_5$ and $M_6$ act on the third block. $M_7$ acts on all qubits in blocks 1 and 2, while $M_7$ acts on all qubit in blocks 2 and 3.

The circuit for determining the syndrome eigenvalues is shown in Fig. [19.11]

We will now see that the $M_i$ have the desired properties:

- They all square to unity (since each of the $Z$’s and $X$’s square to unity and the $X$’s commute amongst each other as do the $Z$’s). Hence their eigenvalues are $\pm 1$.

- They mutually commute. The six $Z$-stabilizers trivially commute with each other as do the two $X$-stabilizers. Comparing the indices on the $Z$-stabilizers with the $X$-stabilizers one sees that either they have none in common (in which case this $X$-stabilizer and $Z$-stabilizer trivially commute) or they have two in common, in which case there are two minus signs when one pulls one of the stabilizers through the other so the overall sign is positive, and again the $X$-stabilizer and the $Z$-stabilizer commute).

- The eigenvalue of the uncorrupted codewords $|\emptyset\rangle$ and $|\bar{1}\rangle$ is $+1$ for all stabilizers.

This is trivially seen for $M_1$–$M_6$ which involve pairs of $Z$ operators, since, for each pair, both qubits are 0 or both are 1 in the codewords. Note that the pairs are entirely within the blocks of three adjacent qubits in Eq. [19.40], see Fig. [19.10]

Next consider $M_7$ and $M_8$ which involve a product of six $X$ operators, each spanning two of the three blocks shown in Fig. [19.10]. For example, $M_7$ is a product of the $X$ operators for the qubits in the first two blocks. We have

$$M_7|\emptyset\rangle = X_1X_2X_3X_4X_5X_6 \frac{1}{2^{3/2}} (|000\rangle + |111\rangle) (|000\rangle + |111\rangle) (|000\rangle + |111\rangle)$$

$$= \frac{1}{2^{3/2}} (|111\rangle + |000\rangle) (|111\rangle + |000\rangle) (|000\rangle + |111\rangle)$$

$$= \frac{1}{2^{3/2}} (|000\rangle + |111\rangle) (|000\rangle + |111\rangle) (|000\rangle + |111\rangle)$$

$$= |\emptyset\rangle,$$  \hspace{1cm} (19.42)

and

$$M_7|\bar{1}\rangle = X_1X_2X_3X_4X_5X_6 \frac{1}{2^{3/2}} (|000\rangle - |111\rangle) (|000\rangle - |111\rangle) (|000\rangle - |111\rangle)$$

$$= \frac{1}{2^{3/2}} (|111\rangle - |000\rangle) (|111\rangle - |000\rangle) (|000\rangle - |111\rangle)$$

$$= \frac{1}{2^{3/2}} (|000\rangle - |111\rangle) (|000\rangle - |111\rangle) (|000\rangle - |111\rangle)$$

$$= |\bar{1}\rangle,$$  \hspace{1cm} (19.43)

so $M_7$ has eigenvalue $+1$ for both uncorrupted codewords. The argument for $M_8$ goes along the same lines.

- The $\pm 1$ eigenvalues of the stabilizers allow one to determine which of the 22 syndromes in Eq. [19.40] the system has projected on to. Recalling the discussion in Sec. [19.3] the eigenvalue is $+1$ if the stabilizer commutes with the operator which caused the 1-qubit corruption, and is $-1$ if it anti-commutes. Each time two different operators on the same qubit are pulled through each other to perform the commutation one generates a minus sign. The operators which generate the corruption are the 21 $X_i$, $Y_i$ and $Z_i$ in Eq. [19.40]. A table of the eigenvalues of the stabilizers for all 22 syndromes is given in Table [19.3]
19.6. CORRECTING ARBITRARY ERRORS: THE SHOR CODE

<table>
<thead>
<tr>
<th>Syndrome</th>
<th>$M_1$</th>
<th>$M_2$</th>
<th>$M_3$</th>
<th>$M_4$</th>
<th>$M_5$</th>
<th>$M_6$</th>
<th>$M_7$</th>
<th>$M_8$</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>+</td>
<td>+</td>
<td>+</td>
<td>+</td>
<td>+</td>
<td>+</td>
<td>+</td>
<td>+</td>
</tr>
<tr>
<td>$X_1$</td>
<td>−</td>
<td>+</td>
<td>+</td>
<td>+</td>
<td>+</td>
<td>+</td>
<td>+</td>
<td>+</td>
</tr>
<tr>
<td>$X_2$</td>
<td>−</td>
<td>−</td>
<td>+</td>
<td>+</td>
<td>+</td>
<td>+</td>
<td>+</td>
<td>+</td>
</tr>
<tr>
<td>$X_3$</td>
<td>+</td>
<td>−</td>
<td>+</td>
<td>+</td>
<td>+</td>
<td>+</td>
<td>+</td>
<td>+</td>
</tr>
<tr>
<td>$X_4$</td>
<td>+</td>
<td>+</td>
<td>−</td>
<td>+</td>
<td>+</td>
<td>+</td>
<td>+</td>
<td>+</td>
</tr>
<tr>
<td>$X_5$</td>
<td>+</td>
<td>+</td>
<td>−</td>
<td>−</td>
<td>+</td>
<td>+</td>
<td>+</td>
<td>+</td>
</tr>
<tr>
<td>$X_6$</td>
<td>+</td>
<td>+</td>
<td>−</td>
<td>−</td>
<td>+</td>
<td>+</td>
<td>+</td>
<td>+</td>
</tr>
<tr>
<td>$X_7$</td>
<td>+</td>
<td>+</td>
<td>+</td>
<td>−</td>
<td>−</td>
<td>+</td>
<td>+</td>
<td>+</td>
</tr>
<tr>
<td>$X_8$</td>
<td>+</td>
<td>+</td>
<td>+</td>
<td>−</td>
<td>−</td>
<td>+</td>
<td>+</td>
<td>+</td>
</tr>
<tr>
<td>$X_9$</td>
<td>+</td>
<td>+</td>
<td>+</td>
<td>−</td>
<td>−</td>
<td>+</td>
<td>+</td>
<td>+</td>
</tr>
<tr>
<td>$Y_1$</td>
<td>−</td>
<td>+</td>
<td>+</td>
<td>+</td>
<td>+</td>
<td>+</td>
<td>−</td>
<td>+</td>
</tr>
<tr>
<td>$Y_2$</td>
<td>−</td>
<td>−</td>
<td>+</td>
<td>+</td>
<td>+</td>
<td>−</td>
<td>+</td>
<td>+</td>
</tr>
<tr>
<td>$Y_3$</td>
<td>+</td>
<td>−</td>
<td>+</td>
<td>+</td>
<td>+</td>
<td>−</td>
<td>+</td>
<td>+</td>
</tr>
<tr>
<td>$Y_4$</td>
<td>+</td>
<td>+</td>
<td>−</td>
<td>+</td>
<td>+</td>
<td>−</td>
<td>+</td>
<td>+</td>
</tr>
<tr>
<td>$Y_5$</td>
<td>+</td>
<td>+</td>
<td>−</td>
<td>−</td>
<td>+</td>
<td>−</td>
<td>+</td>
<td>+</td>
</tr>
<tr>
<td>$Y_6$</td>
<td>+</td>
<td>+</td>
<td>−</td>
<td>−</td>
<td>+</td>
<td>−</td>
<td>+</td>
<td>+</td>
</tr>
<tr>
<td>$Y_7$</td>
<td>+</td>
<td>+</td>
<td>+</td>
<td>−</td>
<td>+</td>
<td>+</td>
<td>−</td>
<td>+</td>
</tr>
<tr>
<td>$Y_8$</td>
<td>+</td>
<td>+</td>
<td>+</td>
<td>−</td>
<td>−</td>
<td>+</td>
<td>+</td>
<td>−</td>
</tr>
<tr>
<td>$Y_9$</td>
<td>+</td>
<td>+</td>
<td>+</td>
<td>+</td>
<td>+</td>
<td>+</td>
<td>+</td>
<td>−</td>
</tr>
<tr>
<td>$Z_1 (= Z_2 = Z_3)$</td>
<td>+</td>
<td>+</td>
<td>+</td>
<td>+</td>
<td>+</td>
<td>+</td>
<td>−</td>
<td>+</td>
</tr>
<tr>
<td>$Z_4 (= Z_5 = Z_6)$</td>
<td>+</td>
<td>+</td>
<td>+</td>
<td>+</td>
<td>+</td>
<td>+</td>
<td>−</td>
<td>+</td>
</tr>
<tr>
<td>$Z_7 (= Z_8 = Z_9)$</td>
<td>+</td>
<td>+</td>
<td>+</td>
<td>+</td>
<td>+</td>
<td>+</td>
<td>+</td>
<td>+</td>
</tr>
</tbody>
</table>

Table 19.3: The eigenvalues of the 8 stabilizers defined in Eq. (19.41) for the 22 syndromes of Shor’s 9-qubit error correcting code. The left column indicates the operator which generates for syndrome from the uncorrupted state. A + sign indicates eigenvalue +1 and a − sign indicates eigenvalue −1. Each stabilizer $M_i$ is measured by an ancilla qubit $x_i$, see Fig. 19.11, such that if $M_i = +1$ then $x_i = 0$ and if $M_i = −1$ then $x_i = 1$. An essential feature is that each of the 22 rows, i.e. syndromes, has a unique pattern of + and − signs.

Let’s make sure that we understand how the syndrome-detection circuit in Fig. 19.11 works. Firstly we remind the reader that if the measurement of an auxiliary qubit, $x_i$ say, is 0, then the value of the corresponding stabilizer $M_i$ is +1, while if the measurement is 1, then the value of $M_i$ is −1. Thus we can say that $x_i$ measures $M_i$, see the discussion of Fig. 19.6 on page 150. Next we discuss how each of the stabilizers works.

- We consider first $M_1$–$M_6$, the stabilizers involving $Z$ operators.
  The ancilla qubits $x_1$ and $x_2$ measure $M_1 = Z_1Z_2$ and $M_2 = Z_2Z_3$ respectively, and so detect a bit-flip error in the first group of three qubits in the 9-qubit encoding of Eq. (19.36), in exactly the same way as for the 3-qubit, bit-flip code shown in Fig. 19.7. Similarly $x_2$ detects a bit-flip error in the second group of three qubits (qubits 4–6), and $x_3$ detects a bit-flip error in the third group of three qubits (qubits 7–9).

- Next we consider $M_7$ and $M_8$, the stabilizers involving $X$ operators.
  The ancilla $x_7$ measures $M_7 = X_1X_2X_3X_4X_5X_6$ and the ancilla $x_8$ measures $M_8 = X_4X_5X_6X_7X_8X_9$. These detect phase flips. $M_7$ acts on the first two groups of three qubits (qubits 1–6) while $M_8$ acts on the second and third groups of three qubits (qubits 4–9).
Let’s suppose that there is a phase flip in one of the qubits in the first group (it doesn’t matter which one; the resulting state is the same). In other words

\[ |\psi\rangle = \alpha|0\rangle + \beta|1\rangle \]  

(19.44)

has been transformed to

\[ |\psi'\rangle = \frac{\alpha}{2^{3/2}} (|000\rangle - |111\rangle) (|000\rangle + |111\rangle) (|000\rangle + |111\rangle) + \frac{\beta}{2^{3/2}} (|000\rangle + |111\rangle) (|000\rangle - |111\rangle) (|000\rangle - |111\rangle). \]  

(19.45)

Acting with the product of the three \(X_i\) in a group has the effect

\[ |000\rangle + |111\rangle \rightarrow |111\rangle + |000\rangle = |000\rangle + |111\rangle \]
\[ |000\rangle - |111\rangle \rightarrow |111\rangle - |000\rangle = - (|000\rangle - |111\rangle). \]  

(19.46)

Recalling that we are considering here a phase flip in the first group, on which \(M_7\) acts but \(M_8\) does not, it follows that

\[ M_7|\psi'\rangle = X_1X_2X_3X_4X_5X_6|\psi'\rangle = -|\psi'\rangle \]
\[ M_8|\psi'\rangle = X_4X_5X_6X_7X_8X_9|\psi'\rangle = |\psi'\rangle. \]  

(19.47)

Hence \(M_7\) has eigenvalue \(-1\) and \(M_8\) has eigenvalue \(+1\). These values are shown in Table 19.3

Similarly, if the phase-flip is in the second group, both \(M_7\) and \(M_8\) have eigenvalue \(-1\) whereas if phase-flip is in the third group, \(M_7\) has eigenvalue \(+1\) and \(M_8\) has eigenvalue \(-1\). These results are also shown in Table 19.3

We now illustrate in more detail how Table 19.3 was obtained by working through a few cases. (Eigenvalues are taken to be \(+1\) unless otherwise stated.)

(a) \(Z_2\): Clearly \(Z_2\) commutes with all the \(Z\)-stabilizers. It anticommutes with \(M_7\) (because it has one qubit in common and \(X\) and \(Z\) anticommute) and commutes with \(M_8\) because it has no qubits in common. Hence \(M_7\) has eigenvalue \(-1\) while all other stabilizers have eigenvalue \(+1\).

(b) \(Z_4\): Both \(M_7\) and \(M_8\) have eigenvalue \(-1\) since they have one qubit in common with \(Z_4\) (and \(X\) and \(Z\) anticommute).

(c) \(X_4\): Clearly \(X_4\) commutes with both \(X\)-stabilizers. It anticommutes with \(M_5\) because it has one qubit in common (and \(Z\) and \(X\) anticommute). Hence \(M_3\) has eigenvalue \(-1\).

(d) \(Y_5\): We note that \(Y\) anticommutes with both \(X\) and \(Z\) so we have to consider all the stabilizers. \(Y_5\) has a qubit in common with \(M_3, M_4, M_7\) and \(M_8\) so these stabilizers have eigenvalue \(-1\).

We recall that each syndrome gives rise to a unique set of \(+1\) and \(-1\) eigenvalues of the stabilizers, see Table 19.3. Thus, measuring the eigenvalues of the eight stabilizers in Eq. \((19.41)\) projects the corrupted state on to one of the 22 syndromes in Eq. \((19.40)\), and the set of eigenvalues determines which one it is. One then applies an appropriate unitary transformation to correct the state if necessary. Note that the Shor code is *explicitly* designed to detect and correct bit-flip (\(X\)) and phase-flip (\(Z\)) errors, but then *automatically* detects and corrects combined bit- and phase-flip (\(ZX \equiv iY\)) errors.

Not only that, it also corrects *arbitrary* errors on a single qubit, which, as discussed in Sec. \(19.5\), can be expressed as *linear combinations* of bit-flip, phase-flip, and combined bit- and phase-flip errors. As an example consider the situation mentioned in Eq. \((19.33)\) in Sec. \(19.5\) in which a qubit has been...
reset to $|0\rangle$. This is an example of a non-unitary operation on the qubit. Let’s take it to be qubit 1 and indicate the codeword qubits by putting the first on the left, the last on the right (we will use the same ordering below for the ancilla qubits). In other words

$$|\psi\rangle = \alpha|0\rangle + \beta|1\rangle$$

has been transformed to

$$|\psi\rangle = \frac{\alpha}{2^{3/2}} (|000\rangle + |011\rangle) (|000\rangle + |111\rangle) (|000\rangle + |111\rangle) + \frac{\beta}{2^{3/2}} (|000\rangle - |011\rangle) (|000\rangle - |111\rangle) (|000\rangle - |111\rangle).$$

According to Eq. (19.33) this can be written as

$$|\psi\rangle = \frac{1}{2} (|1\rangle + X_1 + iY_1 + Z_1)|\psi\rangle,$$

where

$$|\psi\rangle = \alpha (|000\rangle + |111\rangle) (\cdots)_+ (\cdots)_+ + \beta (|000\rangle - |111\rangle) (\cdots)_- (\cdots)_-$$

(19.51a)

$$X_1|\psi\rangle = \alpha (|100\rangle + |011\rangle) (\cdots)_+ (\cdots)_+ + \beta (|100\rangle - |011\rangle) (\cdots)_- (\cdots)_-$$

(19.51b)

$$iY_1|\psi\rangle = \alpha (-|100\rangle + |011\rangle) (\cdots)_+ (\cdots)_+ + \beta (-|100\rangle - |011\rangle) (\cdots)_- (\cdots)_-$$

(19.51c)

$$Z_1|\psi\rangle = \alpha (|000\rangle - |111\rangle) (\cdots)_+ (\cdots)_+ + \beta (|000\rangle + |111\rangle) (\cdots)_- (\cdots)_-,$$

(19.51d)

in which

$$(\cdots)_+ \equiv (|000\rangle + |111\rangle)$$

$$(\cdots)_- \equiv (|000\rangle - |111\rangle).$$

(19.52)

One can verify that adding Eqs. (19.51) (and dividing by 2 according to Eq. (19.50)) does indeed give Eq. (19.49).

Equation (19.50) is the input to the syndrome measurement circuit. According to Table 19.3 after the syndrome measurement circuit in Fig. 19.10 has acted, the state of the system is

$$\frac{1}{2} [ |\psi\rangle |00000000\rangle_A + (X_1|\psi\rangle) |10000000\rangle_A + (iY_1|\psi\rangle) |10000100\rangle_A + (Z_1|\psi\rangle) |00000010\rangle_A ],$$

(19.53)

where $|\cdots\rangle_A$ denotes the ancillas, which are ordered from 1 on the left to 8 on the right. Measuring the ancillas will project the computational qubits on to one of the four syndromes, $|\psi\rangle, X_1|\psi\rangle, iY_1|\psi\rangle, Z_1|\psi\rangle$. Since the measurements of the ancillas tell us which syndrome the state has been projected on to, the computational qubits can then be corrected if necessary.

Thus, Shor’s 9-qubit code, and other codes designed to correct both bit-flip and phase-flip errors, actually correct arbitrary 1-qubit errors. I find this amazing.

### 19.7 Other error-correcting codes

The Shor code uses nine physical qubits to encode one logical qubit. What is the minimum number of physical qubits needed to correct all 1-qubit errors? If we encode using $n$ qubits the dimension of

\[ \text{In footnote 5 we noted that, while a transformation of the combined system+environment is unitary, if the system is coupled to the environment, then a unitary operation applied to system+environment followed by a trace over the environment leaves the system in a new state which is not, in general, related by a unitary transformation to its initial state.} \]
the space of states is \(2^n\). This must be sufficient to contain \(3^n + 1\) mutually orthogonal 2-d subspaces for the syndromes (the 1 is for the uncorrupted state and there are \(n\) possible corruptions with each of the \(X, iY\) or \(Z\) operators). Hence we need

\[
2^n \geq 2(3^n + 1),
\]

so the smallest value is \(n = 5\) which satisfies this condition as an equality.

There is a 5-qubit code, but it turns out to be difficult to construct the necessary gates. A more popular choice is a 7-qubit code due to Steane [Ste96]. The Shor code, which has 9-qubit codewords, is now mainly of pedagogical interest.

We now state, without much discussion, the codewords and stabilizers for the 5-qubit code. Further details are in Mermin [Mer07].

For the 5-qubit code we have \((3 \times 5) + 1 = 16\) mutually orthogonal, two-dimensional subspaces. We therefore need four stabilizers since they each have two eigenvalues \((\pm 1)\) and the number of distinct sets of eigenvalues is \(2^4 = 16\). These stabilizers are

\[
\begin{align*}
M_1 & = Z_2X_3X_4Z_5, \\
M_2 & = Z_3X_4X_5Z_1, \\
M_3 & = Z_4X_5X_1Z_2, \\
M_4 & = Z_5X_1X_2Z_3.
\end{align*}
\]

The circuit to measure the \(M_i\) is shown in Fig. 19.12.

![Figure 19.12: A circuit to measure the error syndrome for the 5-qubit code. The five codeword qubits are at the bottom and the four ancillary qubits at the top. The ancillary qubits determine the values of the four, mutually commuting stabilizers in Eq. (19.55), \(M_1 = Z_2X_3X_4Z_5, M_2 = Z_3X_4X_5Z_1, M_3 = Z_4X_5X_1Z_2, M_4 = Z_5X_1X_2Z_3.\)](image)

The 5-qubit codewords are most conveniently expressed in terms of the \(M_i\):

\[
\begin{align*}
|\bar{0}\rangle & = \frac{1}{4}(1 + M_1)(1 + M_2)(1 + M_3)(1 + M_4)|00000\rangle, \\
|\bar{1}\rangle & = \frac{1}{4}(1 + M_1)(1 + M_2)(1 + M_3)(1 + M_4)|11111\rangle.
\end{align*}
\]

Note that \(|\bar{0}\rangle\) is composed of the 16 basis states with an even number of 1’s, while \(|\bar{1}\rangle\) is composed of the 16 basis states with an odd number of 1’s, so the two codewords are orthogonal. It is not completely trivial to generate these codewords, see Mermin [Mer07] for details.
Table 19.4: The table shows whether the four stabilizers $M_i$ for the 5-qubit error correcting code commute ($+$) or anti-commute ($-$) with the 15 operators $X_i, Y_i$ and $Z_i, i = 1, 2, \cdots, 5$ (which generate a corruption of the codeword) as well as with the identity. Each of the 16 rows has a unique pattern of $+$ and $-$ signs. A $+$ sign corresponds to an eigenvalue $+1$ while a $-$ sign indicates an eigenvalue $-1$.

Furthermore the $M_i$ square to unity, are mutually commuting and each has eigenvalue +1 for the uncorrupted codewords in Eq. (19.56). Each of them commutes or anti-commutes with the $X_i, Y_i$ and $Z_i$ error operators, so the 15 corrupted syndromes and the uncorrupted state are distinguished by the set of $\pm 1$ eigenvalues of the $M$’s, as shown in Table 19.4.

Next I describe briefly the 7-qubit Steane code.

There are 6 stabilizers which are

$$
M_1 = X_1 X_5 X_6 X_7, \quad N_1 = Z_1 Z_5 Z_6 Z_7,
$$

$$
M_2 = X_2 X_4 X_6 X_7, \quad N_2 = Z_2 Z_4 Z_6 Z_7,
$$

$$
M_3 = X_3 X_4 X_5 X_7, \quad N_3 = Z_3 Z_4 Z_5 Z_7. \quad (19.57)
$$

The circuit to detect errors is shown in Fig. 19.13. The 7-qubit codewords are given by

$$
|\bar{0}\rangle = \frac{1}{\sqrt{8}} (1 + M_1)(1 + M_2)(1 + M_3)|0\rangle_7, \quad (19.58)
$$

$$
|\bar{1}\rangle = \frac{1}{\sqrt{8}} (1 + M_1)(1 + M_2)(1 + M_3)|\bar{X}\rangle_7, \quad (19.58)
$$

where

$$
\bar{X} = X_1 X_2 X_3 X_4 X_5 X_6 X_7, \quad (19.59)
$$

so

$$
|1111111\rangle = \bar{X}|0000000\rangle. \quad (19.60)
$$

It is instructive for the student to show the following:

(a) The stabilizers mutually commute and square to the identity.
Figure 19.13: The circuit of Steane’s 7-qubit code to detect errors in the computational qubits, (labeled 1–7 in the figure). There are also six ancilla qubits (at the top) each of which is associated with one of the stabilizers as follows: \( N_1 - N_3 \) correspond to \( x_1 - x_3 \) respectively, and \( M_1 - M_3 \) correspond to \( x_4 - x_6 \) respectively, in the usual way, e.g. \( N_1 = (-1)^x_1 \), \( M_1 = (-1)^x_4 \).

(b) The two states in Eq. (19.58) are orthogonal.

(c) The two states in Eq. (19.58) are normalized.

   Hint: You will need to use that the \( M_i \) square to the identity, as does \( \overline{X} \), and that \( \overline{X} \) commutes with the \( M_i \).

(d) The codewords \( |0\rangle \) and \( |1\rangle \) are eigenstates of each of the stabilizers with eigenvalue +1.

   Hint: Note that \( M_i (1 + M_i) = 1 + M_i \) (why?), that the \( N_j \) commute with \( \overline{X} \) (explain why), and that \( |0\rangle_7 \) is an eigenstate of the \( N_i \) with eigenvalue 1.

A different approach to quantum error correction, but one that seems the most promising, is to use “surface codes” in which the physical qubits are arranged in a square array and the values of the logical qubits are encoded in complicated entangled states of the square array. Unfortunately, I have not been able to find a simple introduction to this topic.

19.8 Fault Tolerant Quantum Computing

So far we have assumed that an error has occurred in some way and that we can correct it by perfect gates which do not introduce any further errors. This is, of course unreasonable since all aspects of quantum computing can introduce errors: acting with gates, measurements, or simply waiting. Looking at the number of gates for Shor’s 9-qubit syndrome-detection code in Fig. 19.11 we might imagine that this circuit could introduce more errors than it corrects. Of particular importance is that the circuit...
19.8. FAULT TOLERANT QUANTUM COMPUTING

165
does not spread an error initially in one qubit into multiple qubits which would then be much harder to correct. A circuit which does not spread errors is said to be “fault tolerant”.

An important result in quantum error correction is the “threshold theorem” which states that if the intrinsic error rate in an individual gate in a fault tolerant circuit is less than a critical value $p_c$, then the overall error rate in the circuit can be reduced to arbitrary low levels by quantum error correction. This means that errors are being corrected faster than they are being generated. However, getting the error rate down to an acceptable level will still require a considerable increase in the number of physical qubits.

Suppose that the intrinsic error rate is $p$ and we have a fault tolerant error correction scheme which corrects 1-qubit errors. This means that the error rate after error correction is $cp^2$ for some constant $c$. If $pc < 1$ then we have decreased the errors, so the threshold error rate is $p_c = 1/c$. How can we go decrease the errors further? Suppose the error correction procedure requires $n$ physical qubits for each logical qubit. We can then take each of the $n$ qubits and error correct these with the same code. This procedure is known as concatenation. We then have $n^2$ physical qubits and the error rate is $c(cp^2)^2 = c^{−1}(cp)^2$. Generalizing, if we concatenate $l$ times, then the number of qubits is $n^l$ while the resulting error rate is $c^{−1}(cp)^{2l}$. Note that while the number of qubits increases exponentially with the level of concatenation $l$, the error rate decreases doubly exponentially with $l$. As an example, to get a feel for what this means, consider the case $p = 1/8, c = 2$, so $cp = 1/4$ and also suppose that $n = 7$ (corresponding to the Steane code). Then successive concatenations give the numbers in Table 19.5.

Table 19.5: Parameters for the concatenation of a fault tolerant circuit with an (artificial) choice of parameters discussed in the text.

<table>
<thead>
<tr>
<th>no. of concatenations ($l$)</th>
<th>error rate (formula)</th>
<th>error rate (numeric)</th>
<th>no. of qubits</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>$p$</td>
<td>$1/2^4 = 0.125$</td>
<td>1</td>
</tr>
<tr>
<td>1</td>
<td>$cp^2 = c^{−1}(cp)^2$</td>
<td>$1/2^5 = 0.03125$</td>
<td>$n$ (= 7)</td>
</tr>
<tr>
<td>2</td>
<td>$c((cp^2)^2)^2 = c^{−1}(cp)^{2^3}$</td>
<td>$1/2^9 = 1.953 \times 10^{-3}$</td>
<td>$n^2$ (= 49)</td>
</tr>
<tr>
<td>3</td>
<td>$c((cp^2)^2)^2 = c^{−1}(cp)^{2^4}$</td>
<td>$1/2^{17} = 7.629 \times 10^{-6}$</td>
<td>$n^3$ (= 343)</td>
</tr>
<tr>
<td>4</td>
<td>$c((cp^2)^2)^2 = c^{−1}(cp)^{2^5}$</td>
<td>$1/2^{33} = 1.164 \times 10^{-10}$</td>
<td>$n^4$ (= 2401)</td>
</tr>
<tr>
<td>5</td>
<td>$c((cp^2)^2)^2 = c^{−1}(cp)^{2^6}$</td>
<td>$1/2^{65} = 2.711 \times 10^{-20}$</td>
<td>$n^5$ (= 16807)</td>
</tr>
</tbody>
</table>

These numbers are not realistic. They correspond to a threshold value of $p_c = 1/c = 1/2$ and any realistic circuit would have a much smaller value. However, they do show, and this is the main point, that the error rate goes down much faster than the number of physical qubits goes up. Of course, the number of physical qubits per logical qubit will still have to be very large to get the error rate down to an acceptable value for computation.

Various calculations have estimated the threshold for 7-qubit Steane code at around $10^{-5}$. To perform error correction one would need individual circuit elements with an error rate significantly less than this, which, to my knowledge, is not feasible at present. Surface codes, which were briefly mentioned above, are estimated to have a higher threshold, of around $10^{-2}$, and it does seem feasible to make gates with a lower error rate than this. For example, at the end of a very long and technical paper, Ref. [FMMC12] estimates that to factor, using Shor’s algorithm, an integer which is too large to be factored on a classical computer (2000 bits), would require no less than around $220 \times 10^6$ qubits with then state-of-the-art superconducting qubits using quantum error correction with surface codes. At present, quantum computers (using the “gate” model of quantum computing which is the topic of this course) have at most a few tens of qubits, so a huge increase in scale will be required. However,

---

7The crucial point is that the new error rate is proportional to the square of the old error rate. I don’t think it’s obvious that one can design a circuit with this property, but a detailed study indicates that one can [NC00, RP14].
who is to say that this cannot happen in a few decades? An example of a comparable increase in scale which has \emph{already} happened is the number of transistors on a modern chip compared with the number on early integrated circuits.

Thus, in my view, in the next few years, we may see quantum computers with a modest number of logical qubits which perform error correction. However, quantum computers with error correction \emph{having enough logical qubits to outperform classical computers} for some useful problem such as integer factorization are for the distant future, if ever.

I thank Eleanor Rieffel for a helpful email exchange on quantum error correction.
Chapter 20

Grover’s Search Algorithm

20.1 Introduction

Grover’s algorithm discussed in this chapter is of a different type from Shor’s algorithm. Whereas Shor (and related algorithms like Simon’s) depend on a quantum Fourier transform (of some sort), Grover’s algorithm involves amplitude amplification.

To motivate Grover’s algorithm consider looking up someone in a phone directory. It is straightforward to lookup a person’s phone number in a directory if one is given the name, because names are in alphabetic order. To locate the name systematically one would go to the midpoint of the list, see which half the name is in, divide that half in two, again see which half the number is in, and so on. One continues this procedure until the size of the region containing the desired entry is just one. For a directory with $N$ entries, this bisection method takes $\log_2 N$ operations (rounded up to the nearest integer if $N$ is not a power of 2) since one halves the range over which the special entry could be at each stage.

By contrast, suppose one is given the number and asked which person has that number. Since the numbers are not ordered, all one can do is go through the entries one at a time and see if each one has the desired name. On average this would take $N/2$ operations before success was achieved.

If $N$ is large this is a huge difference. For example if $N = 10^6$ then $\log_2 N \approx 20$, to be compared with $N/2 = 5 \times 10^5$. Note that the $N$ possible values are represented by the $2^n$ configurations of $N$ qubits so

$$N = 2^n. \quad (20.1)$$

The quantum search algorithm algorithm discussed here, due to Grover, is often presented as such a search of an unstructured database. Grover’s algorithm requires a quantum computer running a subroutine for which the input is a number corresponding to an entry in the database, and which performs a test to see if this is the special value being searched for. For large $N$ it will determine the special value, with probability close to 1, by calling the subroutine only $(\pi/4)\sqrt{N}$ times. This is a quadratic speedup compared with a classical computer. While less spectacular than the exponential speedup of Shor’s algorithm, it can potentially be applied to a wide variety of problems.

1. Though it is doubtful it would ever be used in this way since it would be a very extravagant use of a precious resource to use qubits to store classical information.

2. However, most applications of practical interest have some structure, whereas Grover is designed for problems with no structure. In most cases that Grover could potentially be applied, the structure of the problem allows an efficient classical algorithm which outperforms Grover. Thus it is debated whether the Grover algorithm would be of practical utility, even if one could overcome the severe experimental difficulties of building a large quantum computer.
20.2 The Black Box (Oracle)

To formulate the problem we consider \( n \)-bit integers, one of which, \( a \), is special. The goal is to find \( a \). We need a subroutine which outputs 1 if the input value \( x \) is equal to \( a \) and outputs 0 otherwise, i.e.

\[
\begin{align*}
f(x) &= 0, & (x \neq a), \\
f(a) &= 1. 
\end{align*}
\]

(20.2)

As usual, the function will be determined from a unitary transformation acting on an \( n \)-qubit “input” register and a 1-qubit “output” register which is flipped or not flipped depending on whether \( x \) is the special number \( a \) or not:

\[
\begin{align*}
U |x\rangle_n |y\rangle_1 &= |x\rangle_n |y \oplus f(x)\rangle_1. 
\end{align*}
\]

(20.3)

![Figure 20.1: A black box circuit that executes the first part of a Grover iteration, Eq. (20.3), in which \( f(x) = 0 \) if \( x \neq a \) and \( f(a) = 1 \), for the case of \( n = 5 \) qubits and where the special number \( a \) is 01001. The 6-qubit gate in the center is a five-fold-controlled-NOT gate which acts to flip the target qubit \( y \) only if all the control qubits are 1. The X gates on the left flip qubits \( x_2, x_3 \) and \( x_5 \). Hence the target qubit is flipped if and only if \( x_1 = 1, x_2 = 0, x_3 = 0, x_4 = 1, x_5 = 0 \), which are the bits of \( a \). The X-gates on the right flip back those qubits which had previously been flipped, thus leaving the “input” register, the \( |x_i\rangle \), unchanged. The “output” qubit, \( |y\rangle \), initially, contains information on the function \( f(x) \) in the final state on the right.]

A simple example of such a function for \( n = 5 \) and \( a = 01001 \) is shown in Fig. 20.1. (Recall that \( x_1 \) is the least significant bit.) The target bit is flipped only if all five of the control bits are one, which requires \( x_1 = 1, x_2 = 0, x_3 = 0, x_4 = 1, x_5 = 0 \) (the bits of \( a \)). How to construct such a five-fold-controlled-NOT gate out of 1-qubit and 2-qubit elementary gates is discussed in Mermin [Mer07] §4.2.

Such a black box function is often called an oracle. The oracle can give a yes or no answer as to whether the input is the special number. For the implementation in Fig. 20.1 you might object and say that surely we already know the answer since it is built into the quantum device by placing the X-gates only on those qubits where the special number has a 0 bit. Can’t we just open up the black box and look? In this case the answer is “yes”. However, the implementation of the black box in Fig. 20.1 is just a simple example. The Grover algorithm can also be applied in more useful situations where the value of \( f(x) \) is not built in explicitly but has to be calculated in a non-trivial way. Examples are discussed in Mermin [Mer07] and Nielsen and Chuang [NC00].
It is useful to initially set the “output” bit \( y \) to be 1 and then apply a Hadamard gate before applying \( U \). The “output” bit is then

\[
H|1\rangle = \frac{1}{\sqrt{2}} (|0\rangle - |1\rangle).
\] (20.4)

If the result of \( U \) is \( f(x) = 0 \) then the “output” bit is unchanged. If the result is \( f(x) = 1 \) then \( |0\rangle \rightarrow |1\rangle \) and vice-versa, so the “output” bit changes sign. Consequently

\[
U (|x\rangle_n \otimes H|1\rangle_1) = (-1)^{f(x)}|x\rangle_n \otimes H|1\rangle_1.
\] (20.5)

We can associate the possible sign change with the “input” register in which case the “output” bit remains unchanged. Hence, for simplicity, it will be ignored in what follows. Thus we consider the following unitary operator \( \hat{O} \) acting only on the \( n \)-qubit “input” register:

\[
\hat{O}|x\rangle = (-1)^{f(x)}|x\rangle = \begin{cases} |x\rangle, & x \neq a, \\ -|a\rangle, & x = a. \end{cases}
\] (20.6)

Since \( U \), and hence \( \hat{O} \), are linear, acting with \( \hat{O} \) on a superposition changes the sign of the component along \( |a\rangle \) but leaves the component perpendicular to \( |a\rangle \) unchanged. Hence if

\[
|\psi\rangle = \sum_x c_x|x\rangle,
\] (20.7)

then

\[
|\psi'\rangle \equiv \hat{O}|\psi\rangle = \sum_{x \neq a} c_x|x\rangle - c_a|a\rangle = \sum_x c_x|x\rangle - 2c_a|a\rangle = |\psi\rangle - 2|a\rangle\langle a|\psi\rangle
\] (20.8)

since \( c_a = \langle a|\psi\rangle \). You should check that \( \langle a|\psi'\rangle = -\langle a|\psi\rangle (= -c_a) \) and, for \( x \neq a \), that \( \langle x|\psi'\rangle = \langle x|\psi\rangle (= c_x) \), as required. You should also verify that \( |\psi'\rangle \) is correctly normalized if \( |\psi\rangle \) and \( |a\rangle \) are.

We initialize the \( n \)-qubit input register into a uniform superposition of all basis states by acting with \( n \) Hadamards on \( |0\rangle \):

\[
|\psi_0\rangle = H^\otimes n|0\rangle = \frac{1}{\sqrt{N}} \sum_{x=0}^{N-1} |x\rangle,
\] (20.9)

in which we recall that bits of the \( n \)-digit number \( x \) are the values of the qubits in the computational basis, and also used Eq. (20.1).

It turns out that all the states generated during the Grover algorithm can be written as a linear combination of \( |a\rangle \) and a uniform superposition of all basis states perpendicular to \( |a\rangle \), i.e.

\[
|a_\perp\rangle = \frac{1}{\sqrt{N-1}} \sum_{x \neq a} |x\rangle_n.
\] (20.10)

Hence the states generated by the algorithm can be conveniently drawn as vectors in the 2-dimensional space spanned by these two basis vectors, see Fig. 20.2. The component of the initial state \( |\psi_0\rangle \) in the direction of \( |a\rangle \) is \( 1/2^{n/2} = 1/\sqrt{N} \), so the vector for \( |\psi_0\rangle \) makes an angle \( \theta_0 \) with the \( |a_\perp\rangle \) axis where

\[
\sin \theta_0 = \frac{1}{\sqrt{N}}.
\] (20.11)

Hence we can express \( |\psi_0\rangle \) as

\[
|\psi_0\rangle = \sin \theta_0 |a\rangle + \cos \theta_0 |a_\perp\rangle.
\] (20.12)

\[\text{We omit the subscript } n \text{ on the states from now on since we will only be dealing with } n \text{-qubit states.}\]
Figure 20.2: Projection of the $2^N$-dimensional space on to a 2-dimensional space spanned by $|a\rangle$ and $|a_\perp\rangle$, the latter being a (normalized) equal linear combination of all basis states except for $|a\rangle$ itself, see Eq. (20.10). The vector in bold is the initial state $|\psi_0\rangle$, an equal linear combination of all basis states, see Eq. (20.9). The vector $|\psi_0\rangle$ has a projection $1/\sqrt{N}$ on to $|a\rangle$, so $\sin \theta_0 = 1/\sqrt{N}$, where $\theta_0$ is the angle between $|\psi_0\rangle$ and $|a_\perp\rangle$.

Figure 20.3: Figure showing that the action of the operator $\hat{O}$ is to reflect the state it is acting on, in this case $|\psi_0\rangle$, about the $|a_\perp\rangle$ axis.

As shown in Eq. (20.8) the action of $\hat{O}$ is to invert the component along $|a\rangle$ of the vector it acts on, while keeping the component perpendicular to $|a\rangle$ unchanged. The net effect is to reflect about the $|a_\perp\rangle$ axis. Figure 20.3 shows the effect of $\hat{O}$ on the initial state $|\psi_0\rangle$.

The goal of the Grover algorithm is to iteratively rotate the vector representing the state of the input register from its initial direction, that of $|\psi_0\rangle$ (which is close to the $|a_\perp\rangle$ axis), to a direction close to the $|a\rangle$ axis, because a measurement of it will then give $a$ with a high probability. To perform this rotation we need a second unitary operation that we will discuss in the next subsection.
20.3 The second step of the Grover iteration

The second stage of a single Grover iteration is independent of the special number \(a\). It changes the sign of the component perpendicular to the initial state \(|\psi_0\rangle\) and keeps unchanged the component along \(|\psi_0\rangle\). Denoting this operation by \(\hat{S}\) we have

\[
|\phi\rangle \rightarrow |\phi'\rangle = \hat{S}|\phi\rangle = 2|\psi_0\rangle\langle\psi_0|\phi\rangle - |\phi\rangle,
\]

where \(|\phi\rangle\) is an arbitrary state. You should check that \(\langle\psi_0|\phi'\rangle = \langle\psi_0|\phi\rangle\), so the component along \(|\psi_0\rangle\) is unchanged, and for a state \(|\mu\rangle\) which is orthogonal to \(|\psi_0\rangle\) (i.e. \(\langle\mu|\psi_0\rangle = 0\)), \(\langle\mu|\phi'\rangle = -\langle\mu|\phi\rangle\), showing that the component perpendicular to \(|\psi_0\rangle\) has the sign changed. The net result if to reflect \(|\phi\rangle\) about the direction of \(|\psi_0\rangle\). Figure 20.4 shows the effects of \(\hat{S}\) acting on the state generated by \(\hat{O}|\psi_0\rangle\). The combined effect of \(\hat{O}\) followed by \(\hat{S}\) is to rotate the initial state \(|\psi_0\rangle\) by \(2\theta_0\) in an anti-clockwise direction, i.e. \(2\theta_0\) towards the desired direction of the \(|a\rangle\) axis. The combination of these two operations is called a Grover iteration.

The effect of the first Grover iteration, therefore, is to take the initial state \(|\psi_0\rangle\) and rotate it anti-clockwise by \(2\theta_0\). We will call the resulting state \(|\psi_1\rangle\). It is at an angle \(\theta_1 = \theta_0 + 2\theta_0\) to the \(|a_\perp\rangle\) axis.

20.4 Subsequent iterations

Subsequent Grover iterations perform the same two steps: \(\hat{O}\) which reflects about \(|a_\perp\rangle\) followed by \(\hat{S}\) which reflects about \(|\psi_0\rangle\). The overall circuit implementing the Grover algorithm is shown in Fig. 20.5.

If \(m\) iterations have already been done, so the current state is \(|\psi_m\rangle\). Fig. 20.6 shows the effect of doing an additional iteration. The state \(|\psi_m\rangle\) makes an angle \(\theta_m\) with the \(|a_\perp\rangle\) axis, so \(\hat{O}\) rotates the
direction by $2\theta_m$ clockwise, while $\hat{S}$ rotates it by $2(\theta_m + \theta_0)$ anti-clockwise. The net result is a rotation by $2\theta_0$ (independent of $\theta_m$) anti-clockwise, which is towards the desired direction, $|a\rangle$, i.e.

$$\theta_{m+1} = \theta_m + 2\theta_0,$$

which gives

$$\theta_m = (2m + 1)\theta_0 \quad (20.16)$$

The relationship between $|\psi_m\rangle$, $|a\rangle$ and $|a_\perp\rangle$ is

$$|\psi_m\rangle = \cos \theta_m |a_\perp\rangle + \sin \theta_m |a\rangle. \quad (20.17)$$

Note that $|\psi_m\rangle$, $|a_\perp\rangle$ and $|a\rangle$ are all normalized.

According to Eq. (20.17), the amplitude for $|\psi_m\rangle$ to be measured in state $|a\rangle$, i.e. $\langle a|\psi_m\rangle$, is $\sin \theta_m = \sin[(2m + 1)\theta_0]$, the projection on to the vertical axis in Fig. 20.6. This increases as $m$ increases up to the point where $\theta_m = \pi/2$ but then decreases. One therefore takes the number of Grover iterations, $m$, to be such that $\theta_m \simeq \pi/2$. From Eqs. (20.16) and (20.11) we see that we need

$$\theta_m = (2m + 1)\theta_0 = (2m + 1)\sin^{-1} \frac{1}{\sqrt{N}} = \frac{\pi}{2},$$

which, for large $N$, gives

$$m = \frac{\pi}{4} \sqrt{N}. \quad (20.19)$$

When $\theta_m \simeq \pi/2$ measuring the state gives $a$ with high probability.

We do not have to get the number of iterations precisely right. After $m$ iterations, the probability that a measurement gives $a$ is $\sin^2 \theta_m = \sin^2[(2m + 1)\theta_0]$. Any value of $\theta_m$ in the range

$$\frac{\pi}{4} < \theta_m < \frac{3\pi}{4} \quad (20.20)$$

will get determine $a$ correctly with a probability greater than $1/2$. For large $N$ this corresponds to

$$\frac{\pi}{8} \sqrt{N} < m < \frac{3\pi}{8} \sqrt{N}. \quad (20.21)$$

Note that the probability decreases for $m > (\pi/4)\sqrt{N}$, unlike many algorithms where increasing the number of iterations progressively improves the probability of success.
20.5 Extensions

20.5.1 More than one special value

In the standard implementation of the Grover algorithm it is assumed that there is only one special value. If there are \( M \) solutions, \( a_i, i = 1, \cdots, M \) indicated collectively by the set \( \{a\} \), then, proceeding along the lines of the derivation for one solution, one finds \([NC00, Mer07, Vat16, RP14]\):

(a) The states generated by the Grover algorithm can be written as a linear combination of a uniform
superposition of all the special states,

$$|a⟩ = \frac{1}{\sqrt{M}} \sum_{x \in \{a\}} |x⟩,$$

(20.22)

and a uniform superposition of all the other states,

$$|a⊥⟩ = \frac{1}{\sqrt{N-M}} \sum_{x \notin \{a\}} |x⟩,$$

(20.23)

(b) The initial state, $|ψ⟩_0$, the uniform superposition of all states given in Eq. (20.9), can be written in terms of $|a⟩$ and $|a⊥⟩$ as

$$|ψ⟩_0 = \sqrt{\frac{M}{N}} |a⟩ + \sqrt{\frac{N-M}{N}} |a⊥⟩.$$  

(20.24)

Hence $|ψ⟩_0$ makes an angle $θ_0$ with the $|a⊥⟩$ axis where

$$\sin θ_0 = \sqrt{\frac{M}{N}},$$

(20.25)

rather than Eq. (20.11). Consequently we can write Eq. (20.24) in terms of $θ_0$ in the same way as for $M = 1$, namely Eq. (20.12).

(c) Subsequent iterations rotate the direction of the state by an angle $2θ_0$ towards the $|a⟩$ axis and so, after $m$ iterations, the angle $θ_m$ is given by Eq. (20.16), and the state $|ψ_m⟩$ is given by Eq. (20.17). Hence the effect of each Grover iteration, when expressed in terms of $θ_0$, is the same as for $M = 1$, and the only difference compared with $M = 1$ is that $θ_0$ is given by Eq. (20.25) rather than (20.11).

(d) Assuming $M ≪ N$, then $θ_m$ is approximately $π/2$ when the number of iterations $m$ is given by

$$m = \frac{π}{4} \sqrt{\frac{N}{M}}.$$  

(20.26)

After this number of iterations of the Grover operator, then, with high probability, a measurement of the state will give one of the special values $a_i$ with equal likelihood.

The student is advised to check these steps.

20.5.2 Quantum Counting

The results of the previous subsection are only useful if we know in advance how many special values, $M$, there are. In general we have no prior knowledge of $M$, so how can we determine it? We saw that $G$ rotates vectors in the $|a⟩$–$|a⊥⟩$ plane by an angle $2θ_0$, where $θ_0$ depends on $M$. Hence $G^m$ is a periodic operator with a period $r$ given by $2rθ_0 = 2π$, or, for $M ≪ N$, $r = π\sqrt{N/M}$. Hence if we could find the period of $G^m$ we would know $M$. Fortunately we already learned how to determine the period of a function on a quantum computer, namely the quantum Fourier transform (QFT) introduced by Shor which we described in Chapter 18.

Hence we can determine $M$, and also get one of the special values, by combining the Quantum Fourier Transform with Grover’s algorithm. In fact this “quantum counting” algorithm will even tell you whether or not a special value exists at all, i.e. whether or not $M = 0$. However, quantum counting is more difficult material so details of it will not be covered here, but the interested student can find those details in more advanced texts [NC00, RP14].
Chapter 21

Quantum Protocols Using Photons

There are several problems of interest where qubits can be considered one at a time, without needing any qubit-qubit interactions. Photons are ideal qubits for this because their interactions are immeasurably weak, and they can be propagated down optical fibres for a big distance with little attenuation while preserving their polarization. You will recall that it is the polarization of the photon which characterizes the qubit, e.g.:

\[ |0\rangle \equiv \leftrightarrow, \quad (\text{left—right}) \]
\[ |1\rangle \equiv \downarrow, \quad (\text{up—down}) \]
\[ |+\rangle = H|0\rangle = \frac{1}{\sqrt{2}}(|0\rangle + |1\rangle) \equiv \mathcal{X}, \quad (\text{one of the diagonals}) \]
\[ |-\rangle = H|1\rangle = \frac{1}{\sqrt{2}}(|0\rangle - |1\rangle) \equiv \mathcal{X}, \quad (\text{the other diagonal}). \]

Several quantum protocols involving photons have been successfully implemented. Here we will discuss applications to cryptography and “teleportation”. Some references are [NC00, Vat16, Mer07].

21.1 Quantum Key Distribution

Cryptography is concerned with transmitting secret messages. There are two main approaches:

- **Public Key**
  An example is the RSA scheme which we already met in Chapter 14 in the context of Shor’s algorithm for factoring integers. Let us briefly review the basic idea. Suppose Bob wants to send a message to Alice. Alice sends her public key down an open channel to Bob who uses this to encrypt his message. Alice decodes the encrypted message using her private key. The private key is not shared, only the public key. Security depends on the difficulty of decoding the message without the private key. In the case of RSA we recall that this required factoring a large integer.

- **Private key (or symmetric key).** (Note: public key encryption is not symmetric between sender and receiver.)
  Alice and Bob share a private key, which has been generated and shared in advance. This must be as long as the message and, as we shall explain later, can only be use once. But how do Alice and Bob share the private key securely? Perhaps Alice could put it in a box and send it to Bob by FedEx. This is not convenient which is why internet transactions use public key encryption instead.
We shall now see that quantum mechanics can help with securely sharing private keys, using what is called Quantum Key Distribution (QKD).

The idea of QKD is to create a one-time codepad which Alice and Bob share. By using quantum mechanics, Alice and Bob will be able to detect whether an eavesdropper whom, following tradition, we shall call Eve, is trying to intercept their messages when they share the codepad.

The codepad is a shared random string of bits $R$, which must be at least as long as the message. Alice encodes the message $M$ by bit-wise XOR-ing it with the random string, i.e.

$$M \rightarrow M \oplus R = M'.$$  \hfill (21.2)

Bob decodes the encoded message $M'$ by also XOR-ing it with $R$, i.e.

$$M' \rightarrow M' \oplus R = M.$$  \hfill (21.3)

This works because $M \oplus R \oplus R = M$, as we have discussed several times before in the course.

We now explain why this codepad cannot only be used once securely. Suppose we send two messages using the same codepad, i.e.

$$M_1' = M_1 \oplus R$$
$$M_2' = M_2 \oplus R.$$  \hfill (21.4)

Anyone intercepting the message can XOR the two messages with the result

$$M_1' \oplus M_2' = M_1 \oplus R \oplus M_2 \oplus R = M_1 \oplus M_2,$$  \hfill (21.5)

so the random string has dropped out. Anyone can then use standard methods (e.g. letter frequency) to decrypt. This is harder than for a single message since one has to extract both messages, but feasible. Hence the great security$^1$ coming from using a random bit string has been lost.

How do Alice and Bob know that their random bit string $R$ was not intercepted by Eve as they were sharing it? This is where quantum mechanics comes into play.

We will describe the method proposed by Bennett and Brassard in 1984 (BB84). Alice sends Bob a long string of photons. Each photon is in one of the four polarization states in Eq. (21.1). The polarization states corresponding to qubits $|0\rangle$ and $|1\rangle$ we will call type-Z qubits (since this is the basis in which $Z$ is diagonal). The polarization states corresponding to $H|0\rangle = \frac{1}{\sqrt{2}}(|0\rangle + |1\rangle)$ and $H|1\rangle = \frac{1}{\sqrt{2}}(|0\rangle - |1\rangle)$ we will call basis-X qubits (since this is the basis in which $X$ is diagonal). To decide in which basis to send a photon Alice generates a random integer taking values 0 and 1. If she gets 0 she sends a type-Z photon, and if she gets 1 she sends a type-X photon. Within each basis-type there are two states, which Alice chooses by generating a second random integer, again taking values 0 and 1. If she gets 0 she sends $|0\rangle$ if the Z-basis were chosen and $H|0\rangle$ if the X-basis were chosen. If she gets 1 for the second random number, she sends $|1\rangle$ or $H|1\rangle$, depending on whether the Z-basis or X-basis was chosen. An example of a set of photons sent to Bob is

$$\begin{array}{cccccccc}
\text{basis} & Z & X & X & Z & Z & X & X & \cdots \\
\text{state} & 0 & 1 & 0 & 1 & 1 & 0 & 0 & \cdots 
\end{array}$$  \hfill (21.6)

Bob receives these qubits and decides randomly whether to measure in the Z-basis or the X-basis. Note that the photons are individually identifiable by the sequence in which they arrive.

If the basis in which Alice sends a photon (Z or X) is the same as that in which Bob measures it, then the state which Bob measures, 0 or 1, must be the same as the state that Alice sent. However

$^1$If the bit string is truly random it is impossible to decrypt the message without knowing the string.
if the bases for sending and measuring are different, then Bob will only find the same state as Alice about half the time. Alice tells Bob over an insecure channel which photons were in the $Z$ basis and which in the $X$-basis, but not the state. Bob then tells Alice over an insecure channel for which of the photons he measured in the same basis as she sent it in. They keep these and discard the others (about 1/2 on average).

The onetime codepad is the set of random bits corresponding to the state of the qubits for which Alice and Bob measured in the same basis. Note that this information was not send down the insecure channel.

Let’s complete the above example with a possible set of measurements that Bob made.

```
Alice basis  | Z  | X  | X  | Z  | Z  | X  | Z  | X  | ...
state       | 0* | 0  | 1  | 1  | 0* | 1  | 0  | 0  | ...

Bob basis   | X  | X  | X  | Z  | X  | Z  | Z  | Z  | ...
state       | 1* | 1  | 1  | 1  | 1* | 1  | 0  | 0  | ...
```

(21.7)

For the photons where Alice’s and Bob’s bases agree, the information is boxed. For these photons, the state that Alice generated and that which Bob measured agree. For the other photons, the states agree only half the time on average. In this example, the cases where the states disagree are starred (2 out of the 5 cases where the bases disagree).

Hence the codepad which Alice and Bob have shared is

$$R = 1010 \cdots$$

(21.8)

How can Alice Bob know if Eve is interrupting the photons? Consider the “good” photons, those where Alice and Bob used the same basis. If Eve is not interrupting them, then Alice and Bob agree on the state with 100% probability. However, if Eve measures the photons and sends them on to Bob, then Alice and Bob will have different states some of the time, as we now show. Like Alice and Bob, Eve will have to choose a random basis for each photon. There is probability 1/2 that she will choose a different basis from the common basis of Alice and Bob. Out of these, there is a probability 1/2 that Eve’s intervention will result in her sending on to Bob a photon in the opposite state from the one which Alice sent. Hence, for the bits where Alice and Bob used the same basis, Eve’s intervention results in Alice and Bob having different states about 1/4 of the time.

To see if this is happening, Alice and Bob sacrifice some fraction of the good photons by sending their values for the state down an insecure channel. If about 1/4 of the states disagree, then they know that the photons are being intercepted. If only a small fraction disagree, Alice and Bob would have needed to decide beforehand up to what fraction of disagreements they would consider an acceptable risk in order to still send the message.

In summary, a quantum key distribution protocol is able to detect an eavesdropper because measurements in quantum mechanics change the state in general.

There is a later version, also due to Bennet and Brassard, from 1992 (BB92), in which only two polarizations are used: $\leftrightarrow$ and $\uparrow\downarrow$. Note that these states are not orthogonal. Lack of orthogonality is essential for the method to work.

### 21.2 Homework Problem On Teleportation

Another quantum protocol which has been successfully implemented using photons is teleportation, in which the state of a qubit, though not the physical qubit itself, is transported (teleported) from one location to another. The best reference for this is Mermin [[Mer07].
Here we give an introduction to teleportation in the form of a homework problem.

Suppose that Alice has a qubit in a state

$$|\psi\rangle = \alpha|0\rangle + \beta|1\rangle. \quad (21.9)$$

The values of $\alpha$ and $\beta$ are unknown to her and can not be determined without destroying the superposition. The no-cloning theorem means that we can’t do repeated measurements on copies of this state. This qubit may be the result of a (possibly complicated) quantum computation which Alice would like to send on to Bob to continue the computation. Bob is far away and Alice can not physically transport the qubit to Bob but wants to send the state.

Now Alice and Bob are able to

- share an entangled qubit

$$|\beta_{00}\rangle = \frac{1}{\sqrt{2}} (|0\rangle_a|0\rangle_b + |1\rangle_a|1\rangle_b), \quad (21.10)$$

where $a$ stands for Alice’s qubit and $b$ stands for Bob’s,

- and communicate over a classical channel (e.g. by phone).

Hence, together they have a 3-qubit state,

$$|\phi_0\rangle = \frac{1}{\sqrt{2}} \left( (\alpha|0\rangle_a + \beta|1\rangle_a) \otimes (|0\rangle_a|0\rangle_b + |1\rangle_a|1\rangle_b) \right)$$

$$= \frac{1}{\sqrt{2}} \left( \alpha|000\rangle + \alpha|011\rangle + \beta|100\rangle + \beta|111\rangle \right), \quad (21.11)$$

where the leftmost two qubits refer to Alice and the rightmost qubit to Bob.

Alice now applies a Bell measurement (discussed in Chapter 9) to the two qubits in her possession, see Fig. 21.1.

![Figure 21.1:](image)

(a) Determine the states $|\phi_1\rangle$ and $|\phi_2\rangle$ shown in the figure.

(b) Alice then measures the two qubits in her possession, obtaining results $x$ and $y$ as shown. She then calls up Bob and tells him the result of her measurements.

Explain what Bob needs to do, depending on the results of Alice’s measurements, for his qubit to be in state

$$|\psi\rangle = \alpha|0\rangle_b + \beta|1\rangle_b, \quad (21.13)$$

i.e. the state that was originally in Alice’s possession.
Note:

- The state, but not the physical qubit, has been transported. This is called teleportation.

- This procedure doesn’t violate relativity (according to which information can not be transmitted faster than the speed of light) since classical communication between Alice and Bob is required.

- It does not violate the no-cloning theorem because, at the end, Alice doesn’t have her original state $|\psi\rangle$, only two classical bits $x$ and $y$. There is never more than one copy of $|\psi\rangle$ in existence.

Final Comment:

There are claims that teleportation has been verified experimentally which I will now discuss briefly. One would like to show the following:

- Alice stores state $|\psi\rangle$.

- The state $|\psi\rangle$ is transported to Bob who is far away.

- Bob stores state $|\psi\rangle$.

To transport qubits over a long distance one needs photons. One can teleport photons over a large distance while retaining their polarization, but at present one can not conveniently store them. One can store other types of qubits, e.g. trapped ions, but can’t entangle them over large distances, so they can be teleported only locally. Hence, in my view, a complete demonstration of teleportation, incorporating all three bullet points above, has not yet been achieved.
Bibliography


