INTRODUCTION TO QUANTUM COMPUTING

WITH APPLICATIONS TO FINANCE

OSWALDO ZAPATA, PhD

A Short Introduction to Quantum Computing for Physicists

Oswaldo Zapata

Abstract

These notes provide an introduction to standard topics on quantum computation and communication for those who already have a basic knowledge of quantum mechanics. The main target audience are professional physicists as well as advanced students of physics; however, engineers and computer scientists may also benefit from them.

1 Introduction **2**

2 Quantum Bits **3**
2.1 Classical Bits . . . 4
2.2 Single Qubits . . . 4
2.3 Multiple Qubits . . . 6

3 Quantum Circuits **9**
3.1 Classical Circuit Gates . . . 9
3.2 Single-Qubit Gates . . . 11
3.3 Multiple Single-Qubit Gates . . . 15
3.4 Multi-Qubit Gates . . . 18
3.5 Measurement . . . 28

4 Quantum Algorithms **33**
4.1 Deutsch's Algorithms . . . 34
4.2 Shor's Factoring Algorithm . . . 41
4.3 Superdense Coding and Teleportation . . . 46
4.4 Quantum Simulation . . . 49

5 Quantum Error Correction **53**
5.1 Entanglement with the Environment . . . 54
5.2 Classical Error Correction . . . 57
5.3 Generalities on QEC Codes . . . 59
5.4 Single Qubit Error Correction . . . 60

6 Bibliography **63**

1 Introduction

The goal of the present notes is to introduce the theoretical framework a trained physicist needs to get into quantum computing. Thus, if you are a physicist and you want to learn the basics of quantum computing, these notes are for you. In a matter of hours (maybe dedicating an entire weekend), you will be able to learn all the basics of quantum computer science.

If, as I suppose, you are a physicist, then at some point in your career you took a proper course on quantum mechanics. Of course, I do not assume that you remember everything you studied then, however, I do assume that you already went through all the standard topics as found in the books by Sakurai or Cohen Tannoudji *et al.* This will allow me to focus on aspects of quantum computing that I think are new to you as a physicist. That said, if you think that you forgot most of what you learned about quantum mechanics, you should not worry. Sincerely speaking, the use of quantum mechanics in quantum computing is relatively simple. Moreover, to help you, in general I recall the main physical and mathematical concepts and I provide explicit calculations so you can easily follow what I am explaining.

Quantum computing is usually described as lying at the intersection of quantum mechanics, mathematics and computer science. As I said, I assume that you studied quantum mechanics. Now, concerning mathematics, I am afraid that most physicists are not familiar with the way computer scientists learn the subject. Here I am not referring, of course, to the mathematics used in quantum mechanics, such as linear algebra, but to subjects like formal logic, models of computation or complexity theory. Since I am not an expert in the field, I will simply sketch the main ideas without entering too many details. The interested reader may look at the appropriate literature. Concerning the most basic notions of computer science, such as Boolean algebra and circuits, I assume that you are barely familiar with them (maybe at the level of the first few lines of a Wikipedia article).

The notes are organized as follows: In Section 2, I introduce the quantum systems relevant to quantum computing and review the mathematical formalism necessary to understand them. In Section 3, I describe how these quantum systems can be manipulated and measured. In Section 4, I review some clever ways physicists and computer scientists have found, at least theoretically, to modify the quantum systems in order to compute certain tasks more efficiently than classical computational methods. In Section 5, I explain how the destructive effect of the environment can be reduced so it does not destroy the quantum nature of the system.

A short comment on the organization of these notes. While Sections 2 and 3 must be read one after the other, Sections 4 and 5 are rather independent of each other. So, after reading Sections 2 and 3, read Sections 4 and 5 in the order that suits you.

The Boxes you find within the main text contain additional material that I consider supplementary. Some of them review topics that I assume you already know and some others expand the main text. My recommendation is that while reading these notes, you give a quick glimpse at the Boxes to see what they are about and, depending on your knowledge, read or skip them. If you decide to skip them, you can always come back to them at a later time.

Concerning the Exercises, I have added them to help you understand and become familiar with the subject, not to make you smarter. So, try to do them; they are relatively easy.

I wish to thank my physics friends for reading the notes, suggesting many improvements and, crucially, testing that you can indeed learn from them. I hope they will be helpful to you as well.

I am planning to continue adding new material to these notes; thus, if you have any feedback (maybe you find a typo, you think that I say something that is not completely correct, I ignored a subject or its presentation can be improved, or any other reason you may have), I will sincerely appreciate it if you send me an email to zapata.oswaldo@gmail.com.

Before moving on to the technical details, let me give a very brief overview of the history of the subject. This will allow you to see the content of these notes in perspective.

The first people who thought about the possibility and the necessity of building quantum computers were Yuri Manin (1980) and Richard Feynman (1982). Feynman's vision was more elaborate, and he considered the advantage of a quantum computer over a classical one for simulating complex quantum systems such as molecules. The next important development was the invention by David Deutsch (1985) of the first quantum algorithm with a computational advantage over classical models of computation. Almost a decade later, there was the discovery by Peter Schor (1985) that quantum computers may be more efficient at solving the prime factorization problem, a scheme widely used to secure the transmission of data. A couple of years later Lov Grover (1996) created and proved that his quantum algorithm for finding an element in a large set of data was more efficient than any possible classical algorithm. The last breakthrough we want to mention is the discovery, also by Peter Shor (1995), that quantum information can indeed be protected against the pernicious effects of the environment.

Look at the Bibliography or popular science literature for more on the history of quantum computing.

2 Quantum Bits

A *computer* is a physical device that, when supplied with the correct set of data, generally known as the *input*, provides another set of data, the *output*. From this general definition it follows that despite our familiarity with modern personal computers, a computer is not necessarily an electronic device. Actually, the first computer conceived and built under the supervision of Charles Babbage in the 19th century was a purely mechanical device with no electronics in it.

If you think for a moment about this wide-ranging definition, you will quickly realize that there are infinite different ways we can write (encode) the initial message we want to communicate to the computer. Ultimately, the way we should encode it will depend on the language spoken by the device, that is, the system of words and rules used by the computer to operate. As with human language, the basic elements of the language of the computer are the words and characters used to construct it.

To make the transition from classical to quantum information processing as smooth as possible, we will start reviewing the basics of classical information theory. Then, we will concentrate on the quantum case.

2.1 Classical Bits

As you certainly already know, the language spoken by ordinary computers is the *binary system*. The latter assumes that every piece of information, for example, a number, a letter or a color, has a unique expression as a finite sequence of zeros and ones. In the binary system, the number 39 is written 100111. Sometimes, by convention, the sequence 01000001 is assigned to the letter A and 11111111 00000000 00000000 to the color red.

These sequences of zeros and ones are called *bit strings* and are somehow equivalent to the words used by humans. Each individual digit of a binary string is called a *bit* (from *bi*nary dig*it*) and is the most basic piece of classical information. This is the analog of the letters used in alphabetic languages. The number of bits in a bit string is known as the *size* of the string.

Here we will only be interested in the binary system applied to numbers. If you are given a positive integer number N in the usual decimal system, the corresponding binary string will be given by the following formula,

$$N = 2^{n-1}b_1 + 2^{n-2}b_2 + \ldots + 2^0 b_n \longleftrightarrow b_1\, b_2\, \ldots\, b_n\,. \tag{2.1}$$

For example,

$$39 = 2^5 b_1 + 2^4 b_2 + 2^3 b_3 + 2^2 b_4 + 2^1 b_5 + 2^0 b_6 = 2^5 1 + 2^4 0 + 2^3 0 + 2^2 1 + 2^1 1 + 2^0 1\,,$$

thus,

$$39 \longleftrightarrow 100111\,.$$

Exercise 2.1. Write the bit string equivalent to every natural number from 1 to 20.

Exercise 2.2. Express 56 and 83 in binary notation.

2.2 Single Qubits

The words a quantum computer understands, that is, the carriers of information, are called *quantum bits* or *qubits*, for short. The simplest piece of quantum information is the *single qubit* or 1 *qubit*. It is a two-level quantum system described by a complex two-dimensional unit state vector

$$|\psi\rangle = a|\varphi_1\rangle + b|\varphi_2\rangle\,, \tag{2.2}$$

where a and b are complex numbers, $a, b \in \mathbb{C}^2$, and the vectors $|\varphi_1\rangle$ and $|\varphi_2\rangle$ are two arbitrary orthonormal vectors spanning the Hilbert space $\mathcal{H} \cong \mathbb{C}^2$ where the qubit $|\psi\rangle$ lives. The real number $|a|^2$ is the probability of measuring the system in the state $|\varphi_1\rangle$ and $|b|^2$ the probability of measuring it in $|\varphi_2\rangle$. Of course, since the only possible outcomes of a measurement are $|\varphi_1\rangle$ and $|\varphi_2\rangle$, it follows that $|a|^2 + |b|^2 = 1$. I remind you that the basis vectors $|\varphi_1\rangle$ and $|\varphi_2\rangle$ are chosen to be orthonormal, that is, $\langle\varphi_r|\varphi_s\rangle = \delta_{rs}$, where $r, s = 1, 2$, because we want the two states to be perfectly distinguishable. The symbol $\langle\ |\ \rangle$, of course, indicates the inner product on the Hilbert space $\mathcal{H}$.

Exercise 2.3. How is the inner product on a Hilbert space usually defined?

If you are the sort of person that prefers to have a physical picture in mind, you may think of a qubit as an electron with two possible spins, a spin up $|\uparrow\rangle$ and a spin down $|\downarrow\rangle$, a photon with a vertical $|\uparrow\rangle$ and a horizontal $|\rightarrow\rangle$ polarization, or an atom with two energy level states $|E_0\rangle$ and $|E_1\rangle$. We will not use explicitly any of these physical representations; however, at times it can be handy to have these pictures in mind. This is somehow analogous to the correspondence made in classical circuit theory between the binary values 0 and 1 and a zero or non-zero voltage, respectively, along a piece of wire. In both cases, classical and quantum, a purely theoretical discussion can be carried out without paying attention to any of these real implementations. This is the approach we will take in these notes.

Even though you already studied most of the quantum mechanics used in quantum computing, there are various conventions and original points of view that are worth following. To begin, we will express the state vector of a single qubit as follows,

$$|q\rangle = \alpha_0|0\rangle + \alpha_1|1\rangle\,. \tag{2.3}$$

The notation $|q\rangle$ is unconventional. In fact, as usual in quantum mechanics, most authors use $|\psi\rangle$. However, we follow the standard convention employed in quantum computing and denote the orthonormal basis vectors by $|0\rangle$ and $|1\rangle$ to emphasize the similitude with the classical binary system. The set $\{|0\rangle, |1\rangle\}$ is known as the *computational basis*. If a state vector, say $|i\rangle$, can only take the values $|0\rangle$ or $|1\rangle$, it is usual to simplify the notation by writing $i \in \{0, 1\}$ or $i = 0, 1$ instead of $|i\rangle \in \{|0\rangle, |1\rangle\}$. Notice that in our notation, if $i, j = 0, 1$, then $\langle i|j\rangle = \delta_{ij}$. We will use $\mathcal{H}_q \cong \mathbb{C}^2$ to refer to the Hilbert space of a single qubit.

Another useful set of orthonormal vectors in the Hilbert space of a single qubit $\mathcal{H}_q$ is the so called *Hadamard basis* $\{|+\rangle, |-\rangle\}$. The latter is given in terms of the computational basis vectors by

$$|+\rangle = \frac{1}{\sqrt{2}}|0\rangle + \frac{1}{\sqrt{2}}|1\rangle\,, \qquad |-\rangle = \frac{1}{\sqrt{2}}|0\rangle - \frac{1}{\sqrt{2}}|1\rangle\,. \tag{2.4}$$

The converse relations are

$$|0\rangle = \frac{1}{\sqrt{2}}|+\rangle + \frac{1}{\sqrt{2}}|-\rangle\,, \qquad |1\rangle = \frac{1}{\sqrt{2}}|+\rangle - \frac{1}{\sqrt{2}}|-\rangle\,. \tag{2.5}$$

The state vector $|q\rangle$ of a single qubit can then be rewritten as $|q\rangle_H = \alpha_+|+\rangle + \alpha_-|-\rangle$, where

$$\alpha_+ = \frac{\alpha_0 + \alpha_1}{\sqrt{2}}\,, \qquad \alpha_- = \frac{\alpha_0 - \alpha_1}{\sqrt{2}}\,. \tag{2.6}$$

According to definition (2.3), the state vector of a single qubit is a function of the two complex numbers α_0 and α_1. That is, we can write more explicitly

$$|q(\alpha_0, \alpha_1)\rangle = \alpha_0|0\rangle + \alpha_1|1\rangle\,. \tag{2.7}$$

Now, since two complex numbers are equivalent to four real numbers and the normalization condition imposes that $|\alpha_0|^2 + |\alpha_1|^2 = 1$, these four numbers reduce to three. Additionally, since two state vectors that differ by a global phase, in fact represent the same physical system, the three real numbers finally reduce to two. The new variables, that we denote θ and ϕ, with $\theta \in [0, \pi]$ and $\phi \in [0, 2\pi)$, can be chosen so that

$$|\alpha_0| = \cos(\theta/2)\,, \qquad |\alpha_1| = \sin(\theta/2)\,. \tag{2.8}$$

Note that $|\alpha_0|^2 + |\alpha_1|^2$ is still equal to 1.

Exercise 2.4. Complete the missing steps.

The general expression of the single-qubit state vector in these new variables is

$$|q(\theta,\phi)\rangle = \cos(\theta/2)|0\rangle + e^{i\phi}\sin(\theta/2)|1\rangle\,. \tag{2.9}$$

In particular,

$$|q(0,\phi)\rangle = |0\rangle\,, \qquad |q(\pi,\phi)\rangle = |1\rangle\,. \tag{2.10}$$

We also have,

$$|q(\pi/2,0)\rangle = \frac{1}{\sqrt{2}}|0\rangle + \frac{1}{\sqrt{2}}|1\rangle = |+\rangle\,, \tag{2.11}$$

and

$$|q(\pi/2,\pi)\rangle = \frac{1}{\sqrt{2}}|0\rangle - \frac{1}{\sqrt{2}}|1\rangle = |-\rangle\,. \tag{2.12}$$

This parametrization of the state vector of a single qubit has a useful visual representation. Suppose that the variables θ and ϕ are the usual spherical coordinates. Then, the state vector of a qubit will be represented by a point — or arrow — on the unit sphere. For example, the north pole corresponds to the basis state vector $|0\rangle$ and the south pole to $|1\rangle$. This unit sphere is called the *Bloch sphere*.

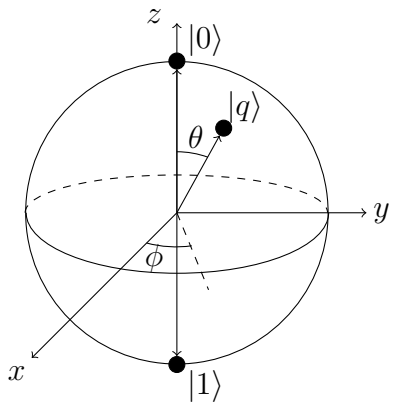

Fig. 1. The Bloch sphere.

Exercise 2.5. What is the position of the Hadamard basis vectors $|+\rangle$ and $|-\rangle$ in the Bloch sphere?

Exercise 2.6. Show that orthogonal states are anti-parallel in the Bloch sphere.

2.3 Multiple Qubits

If a single qubit is a quantum system whose state vector lives in a two-dimensional complex Hilbert space, $|q\rangle = |q_1\rangle \in \mathcal{H}_{q_1} \cong \mathbb{C}^2$, a *2 qubit* is a quantum system whose state vector lives in a Hilbert space which is the tensor product of the Hilbert spaces of two single qubits, $|q_2\rangle \in \mathcal{H}_{q_2} = \mathcal{H}_{q_1} \otimes \mathcal{H}_{q_1'} \cong \mathbb{C}^{2^2}$.

In order to have a clean notation for higher qubits, we will rewrite the state vector of a single qubit as follows,

$$|q_1\rangle = \alpha_0|0\rangle + \alpha_1|1\rangle = \sum_{i=0,1} \alpha_i|i\rangle\ . \tag{2.13}$$

Following the same notation, the state vector of a 2 qubit is simply

$$|q_2\rangle = \alpha_{00}|0\,0\rangle + \alpha_{01}|0\,1\rangle + \alpha_{10}|1\,0\rangle + \alpha_{11}|1\,1\rangle = \sum_{i,j} \alpha_{ij}|ij\rangle\,, \tag{2.14}$$

where $i, j = 0, 1$. From now on, to avoid cluttering the formulas, we will assume that — unless otherwise indicated — the indices i, j, k under the summation symbol take the values 0 and 1. By convention, the first element in the ket $|i\,j\rangle$ represents a computational basis vector of $\mathcal{H}_{q_1}$ and the second element a basis vector of $\mathcal{H}_{q_1'}$. Thus, $\langle i\,j|k\,l\rangle = \langle i|k\rangle\langle j|l\rangle = \delta_{ik}\delta_{jl}$. The mutually orthonormal states $|00\rangle, |01\rangle, |10\rangle$ and $|11\rangle$ form the *computational basis* of $\mathcal{H}_{q_2}$.

Exercise 2.7. Do you remember how the inner product on $\mathcal{H}_{q_2}$ is given in terms of the inner products on the individual Hilbert spaces $\mathcal{H}_{q_1}$ and $\mathcal{H}_{q_1'}$?

Note: If you had problems understanding the beginning of this section, I recommend you to read the following Box. It summarizes the main mathematical concepts and conventions we will use to describe multiple qubits. If you understood everything, then you can confidently skip it.

Box 2.1. Tensor product spaces.

A 2 qubit, simply put, is the composite system of two single qubits. Here we must remember that, since the single qubits can interact between them, the complete description of the whole 2 qubit system may contain information that is not available at the level of the individual qubits.

The Hilbert space $\mathcal{H}_{q_2}$ of the composite system is given by the *tensor product* of the two individual Hilbert spaces,

$$\mathcal{H}_{q_2} = \mathcal{H}_q \otimes \mathcal{H}_{q'}\,. \tag{2.15}$$

This means the following: given the single qubits $|q\rangle, |\tilde{q}\rangle \in \mathcal{H}_q$ and $|q'\rangle, |\tilde{q}'\rangle \in \mathcal{H}_{q'}$, the *tensor product of two vectors* is a map

$$\otimes\colon \mathcal{H}_q \otimes \mathcal{H}_{q'} \to \mathcal{H}_q \otimes \mathcal{H}_{q'}\,, \tag{2.16}$$

which satisfies

$$c(|q\rangle \otimes |q'\rangle) = (c|q\rangle) \otimes |q'\rangle = |q\rangle \otimes (c|q'\rangle)\,, \tag{2.17}$$

for every complex constant c, and

$$|q\rangle \otimes (|q'\rangle + |\tilde{q}'\rangle) = |q\rangle \otimes |q'\rangle + |q\rangle \otimes |\tilde{q}'\rangle\,, \tag{2.18}$$

$$(|q\rangle + |\tilde{q}\rangle) \otimes (|q'\rangle) = |q\rangle \otimes |q'\rangle + |\tilde{q}\rangle \otimes |q'\rangle\,. \tag{2.19}$$

We can use this definition of the tensor product between vectors to define the tensor product between entire Hilbert spaces.

If $\{|0\rangle, |1\rangle\}$ is a basis for $\mathcal{H}_q$ and $\{|0'\rangle, |1'\rangle\}$, is a basis for $\mathcal{H}_{q'}$, the tensor product of these basis vectors, that is, $|0\rangle \otimes |0'\rangle$, $|0\rangle \otimes |1'\rangle$, $|1\rangle \otimes |0'\rangle$ and $|1\rangle \otimes |1'\rangle$ are basis vectors for $\mathcal{H}_{q_2} = \mathcal{H}_q \otimes \mathcal{H}_{q'}$. In other words, every element $|q_2\rangle$ in $\mathcal{H}_{q_2}$ has a unique expression of the form

$$|q_2\rangle = \sum_{i,j'} \alpha_{ij'}|i\rangle \otimes |j'\rangle\,, \tag{2.20}$$

where the coefficients $\alpha_{ij'}$ are complex numbers. Often, to lighten the notation, one drops the symbol $\otimes$ between the vectors. Additionally, one simply writes $|j\rangle$ instead of $|j'\rangle$ because it is clear that the second basis vector is in $\mathcal{H}_{q'}$. Thus,

$$\mathcal{H}_{q_2} \ni |q_2\rangle = \sum_{i,j} \alpha_{ij} |i\rangle|j\rangle \in \mathcal{H}_q \otimes \mathcal{H}_{q'} \,. \tag{2.21}$$

A further simplification is to write $|i\, j\rangle$ instead of $|i\rangle|j\rangle$:

$$|q_2\rangle = \sum_{i,j} \alpha_{ij} |i\, j\rangle \,. \tag{2.22}$$

The inner product on the Hilbert space $\mathcal{H}_{q_2}$ is related to the inner products on $\mathcal{H}_q$ and $\mathcal{H}_{q'}$ by the following formula,

$$\begin{aligned} \langle q_2 | q_2' \rangle &= \Big(\sum_{i,j} \alpha_{ij} |i\, j\rangle, \sum_{k,l} \alpha'_{kl} |k\, l\rangle \Big) = \sum_{i,j,k,l} \alpha^*_{ij} \alpha'_{kl} \langle i\, j | k\, l\rangle \\ &= \sum_{i,j,k,l} \alpha^*_{ij} \alpha'_{kl} \langle i|k\rangle \langle j|l\rangle = \sum_{i,j} \alpha^*_{ij} \alpha'_{ij} \,. \end{aligned} \tag{2.23}$$

Exercise 2.8. How would you define the tensor product between n single-qubit Hilbert spaces?

Remember that composite quantum systems, such as 2 qubits, can be *entangled*, namely, can be in a physical state whose corresponding vector cannot be written as the tensor product of single qubits. In other words, an entangled state is not a *product state*. What we mean by this is the following: if we multiply two single qubits,

$$\begin{aligned} |q\rangle|q'\rangle &= \big(\alpha_0|0\rangle + \alpha_1|1\rangle\big)\big(\alpha_{0'}|0'\rangle + \alpha_{1'}|1'\rangle\big) \\ &= \alpha_0\alpha'_0|0\,0\rangle + \alpha_0\alpha'_1|0\,1\rangle + \alpha_1\alpha'_0|1\,0\rangle + \alpha_1\alpha'_1|1\,1\rangle \\ &= \sum_{i,j} \alpha_i \alpha'_j |i\, j\rangle \,, \end{aligned} \tag{2.24}$$

the entangled states in $\mathcal{H}_{q_2}$ are those for which $\alpha_{ij} \neq \alpha_i \alpha'_j$.

Entangled states are a purely quantum phenomenon. They generally result from the interaction of two or more quantum systems.

Exercise 2.9. Convince yourself that $1/\sqrt{2}(|0\,0\rangle + |1\,1\rangle)$ is an entangled state.

For 3 qubits, the definition is similar: $|q_3\rangle \in \mathcal{H}_{q_3} = \mathcal{H}_{q'_1} \otimes \mathcal{H}_{q''_1} \otimes \mathcal{H}_{q'''_1} \cong \mathbb{C}^{2^3}$. In the computational basis $\{|i\, j\, k\rangle\}$ of $\mathcal{H}_{q_3}$,

$$|q_3\rangle = \sum_{i,j,k} \alpha_{ijk} |i\, j\, k\rangle \,. \tag{2.25}$$

Exercise 2.10. What condition is satisfied by the entangled states in $\mathcal{H}_{q_3}$?

Exercise 2.11. Does the 3-qubit state vector $1/\sqrt{2}(|0\,0\,0\rangle + |1\,1\,1\rangle)$, known as the *GHZ state*, represents an entangled system?

The generalization to n qubits is straightforward. A *multiple qubit* or n *qubit*, for $n \geq 2$, is a quantum system whose state vector $|q_n\rangle \in \mathcal{H}_{q_n} = \mathcal{H}_{q'_1} \otimes \ldots \otimes \mathcal{H}_{q'_1{}^n} \cong \mathbb{C}^{2^n}$. We will often use the notation $|Q\rangle = |q_n\rangle$ and $\mathcal{H}_Q = \mathcal{H}_{q_n}$. In the *computational basis* $\{|i_1 \ldots i_n\rangle\}$ of $\mathcal{H}_Q$, the multiple qubit state vector $|Q\rangle$ is given by the linear combination

$$|Q\rangle = \sum_{i_1,\ldots,i_n} \alpha_{i_1\ldots i_n} |i_1 \ldots i_n\rangle \,, \tag{2.26}$$

where the coefficients $\alpha_{i_1\ldots i_n}$ are complex numbers.

Exercise 2.12. What is the condition satisfied by the entangled states in $\mathcal{H}_Q$?

To simplify the notation further, usually the bit string $i_1 \ldots i_n$ appearing in the state vector $|i_1 \ldots i_n\rangle$ is expressed in decimal notation using (2.1),

$$|Q\rangle = \sum_{x=0}^{2^n-1} \alpha_x |x\rangle \,. \tag{2.27}$$

For example, a 2 qubit can alternatively be written in binary or decimal notation

$$\begin{aligned} |q_2\rangle &= \alpha_{00}|0\,0\rangle + \alpha_{01}|0\,1\rangle + \alpha_{10}|1\,0\rangle + \alpha_{11}|1\,1\rangle \\ &= \alpha_0|0\rangle + \alpha_1|1\rangle + \alpha_2|2\rangle + \alpha_3|3\rangle \,. \end{aligned} \tag{2.28}$$

Even though the first two terms in the last line look exactly the same as the definition (2.3) of a single qubit state vector, there is no risk of confusion because the context will always clearly indicate the one we will be dealing with.

3 Quantum Circuits

Before we start building a computer, we need to decide in advance what sort of tasks it will perform and find the most efficient way of achieving them. Later on we will have time to come back to the concept of efficiency in computer science. However, let us give you an intuitive idea. Suppose we have to automatically generate and tabulate the values of a given polynomial function between two real numbers. To do this, we can use Babbage's "Difference Engine," a heavy, slow and expensive mechanical device. In principle, there is nothing wrong with it. However, I think we all agree that today this is not the most efficient way of performing our tasks. That is, it is not enough to come up with clever theoretical ideas; these ideas must be transformable into practical devices that can process information efficiently. This interplay between theoretical and practical aspects is key in computer science. It was the invention of the transistor in 1947 that consolidated the classical circuit model of computation and gave rise to modern computers. We start this section with a brief overview of digital circuits to better understand how quantum computing relies on, but also goes beyond this classical model.

3.1 Classical Circuit Gates

As we said, an ordinary digital computer understands the binary language of zeros and ones. We provide our computer with a string of zeros and ones (the input), it

processes them and at the end it delivers a new string of zeros and ones (the output). This process, which can be mechanical, electric, or of any other physical nature, is in general expressed mathematically by a function f from the space of bit strings of size l to the space of bit strings of size m, $f\colon \{0,1\}^l \to \{0,1\}^m$. These functions are called *(vector-valued) Boolean functions*. Here we are interested in these functions, that is, in the way the device processes information.

Computer science is a subject that, at least as we approach it here, is at its core in part theoretical and in part practical. Let us say we have a Boolean function $f\colon \{0,1\}^l \to \{0,1\}^m$ and we want to build a device that performs the same operation as f. How should we proceed? Theoretical computer scientists have arrived at the conclusion that any binary function f, no matter how difficult it is, can always be reconstructed by using a combination of functions that are actually easier to materialize in the real world. These more elementary functions are called *elementary* or *basic logic gates*. This is the essence of the *classical circuit model of computation*.

The *NOT gate* is one of these classical basic functions,

$$\mathrm{NOT}\colon \{0,1\} \to \{0,1\}\,, \qquad b \mapsto \mathrm{NOT}(b) = \bar{b}\,. \tag{3.1}$$

The bar over the letter b denotes the logic negation of the bit b. In simple words, if the input is 0, then the output is 1, and vice versa. We can also represent the action of the NOT gate on a bit as follows,

$$0 \overset{\mathrm{NOT}}{\longmapsto} 1\,, \qquad 1 \overset{\mathrm{NOT}}{\longmapsto} 0\,. \tag{3.2}$$

The next basic gate is the *OR gate*,

$$\mathrm{OR}\colon \{0,1\}^2 \to \{0,1\}\,, \qquad b_1 b_2 \mapsto \mathrm{OR}(b_1 b_2)\,, \tag{3.3}$$

given explicitly by,

$$0\,0 \overset{\mathrm{OR}}{\longmapsto} 0\,, \qquad 0\,1 \overset{\mathrm{OR}}{\longmapsto} 1\,, \qquad 1\,0 \overset{\mathrm{OR}}{\longmapsto} 1\,, \qquad 1\,1 \overset{\mathrm{OR}}{\longmapsto} 1\,. \tag{3.4}$$

Note that, in contrast to the NOT gate, the input of an OR gate is a string of size 2. So, we call it a 2-bit gate. The last basic gate on our list is the *AND gate*,

$$\mathrm{AND}\colon \{0,1\}^2 \to \{0,1\}\,, \qquad b_1 b_2 \mapsto \mathrm{AND}(b_1 b_2)\,, \tag{3.5}$$

which transforms

$$0\,0 \overset{\mathrm{AND}}{\longmapsto} 0\,, \qquad 0\,1 \overset{\mathrm{AND}}{\longmapsto} 0\,, \qquad 1\,0 \overset{\mathrm{AND}}{\longmapsto} 0\,, \qquad 1\,1 \overset{\mathrm{AND}}{\longmapsto} 1\,. \tag{3.6}$$

The result we referred above establishes that any Boolean function $f\colon \{0,1\}^l \to \{0,1\}^m$ can be expressed as a composition of these elementary gates. It is then said that the gates NOT, OR and AND form a *universal set of (classical) (logic) gates*.

Just as every component of an electric circuit has a visual representation, the three electronic gates just mentioned have also a corresponding *circuit diagram*,

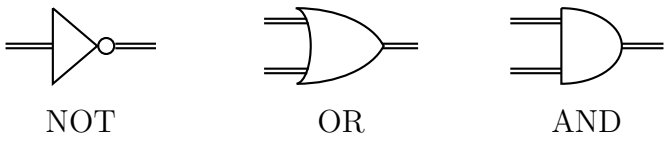

Fig. 2. Three classical basic electronic gates.

By convention, the inputs enter from the left of the gate and the outputs exist from the right. The double lines represent the wires through which the data, namely, the bit strings, flow to go from one gate to the next. A classical circuit, which, as we said, can always be made using only the NOT, OR and AND gates, will consequently have an associated visual representation, in general a convoluted circuit diagram, showing every single element necessary to build it and the relative position between them.

3.2 Single-Qubit Gates

As well as every Boolean function can be thought of as a concatenation of elementary logic gates, we will see that any unitary transformation on a qubit can be decomposed into a sequence of elementary quantum gates.

As you know, according to quantum mechanics, the evolution of a quantum system is given by the action of a unitary operator on the state vector that describes the system at some moment in time. That is, if our quantum system is an n qubit, it will evolve from its initial state $|Q_0\rangle$ to its final state $|Q_f\rangle$ according to $|Q_0\rangle \overset{U}{\mapsto} |Q_f\rangle = U|Q_0\rangle$. In quantum computing, unitary transformations acting on qubit state vectors, especially when the number of qubits is small, are also called *(quantum logic) gates* or *unitaries*. In this subsection we will only deal with unitaries on single qubits.

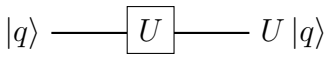

Fig. 3. Circuit diagram of a single-qubit gate.

As for classical circuits, the qubits move from left to right. However, notice that we use single lines to represent the quantum communication channels (to distinguish them from the double lines we used above for classical wires).

Exercise 3.1. Why quantum transformations must be unitary, $U^{-1} = U^\dagger$?

Because the Hilbert space of a single qubit is a 2-dimensional vector space, it is usual to express the computational basis vectors in column vector notation,

$$|0\rangle = \begin{bmatrix} 1 \\ 0 \end{bmatrix} , \qquad |1\rangle = \begin{bmatrix} 0 \\ 1 \end{bmatrix} . \tag{3.7}$$

Exercise 3.2. Show that the matrices assigned to the computational basis vectors are indeed consistent with the orthonormality condition we imposed on them.

With this choice, the state vector of the single qubit (2.3) has the column vector form

$$|q\rangle = \alpha_0 \begin{bmatrix} 1 \\ 0 \end{bmatrix} + \alpha_1 \begin{bmatrix} 0 \\ 1 \end{bmatrix} = \begin{bmatrix} \alpha_0 \\ \alpha_1 \end{bmatrix} . \tag{3.8}$$

Correspondingly, its evolution will be determined by a *single-qubit gate* represented by a 2×2 matrix

$$U = \begin{bmatrix} U_{00} & U_{01} \\ U_{10} & U_{11} \end{bmatrix} . \tag{3.9}$$

Then, when the single qubit $|q\rangle$ enters the gate U, on the other side of the gate exists a state

$$U|q\rangle = \begin{bmatrix} U_{00}\alpha_0 + U_{01}\alpha_1 \\ U_{10}\alpha_0 + U_{11}\alpha_1 \end{bmatrix} . \tag{3.10}$$

Exercise 3.3. Show that in index notation

$$U|0\rangle = \sum_i U_{i0}|i\rangle\,, \qquad U|1\rangle = \sum_i U_{i1}|i\rangle\,, \tag{3.11}$$

and thus, more generally,

$$U|q\rangle = \sum_{i,j} \alpha_j U_{ij}|i\rangle\,. \tag{3.12}$$

If we are not given the explicit matrix representation of the single-qubit gate as in (3.9), but only its action on the computational basis vectors, the single-qubit gate is abstractly given by the ket-bra expression

$$U = \sum_{i,j} U_{ij}|i\rangle\langle j|\,. \tag{3.13}$$

From here, we can find the matrix by using the following formula:

$$U = \begin{bmatrix} \langle 0|U|0\rangle & \langle 0|U|1\rangle \\ \langle 1|U|0\rangle & \langle 1|U|1\rangle \end{bmatrix} . \tag{3.14}$$

That is, the elements of a 2×2 matrix associated to a single-qubit gate are given by

$$U_{ij} = \langle i|U|j\rangle\,. \tag{3.15}$$

If a single qubit enters two gates, first U_1 and then U_2, quantum mechanics tells us that the outgoing qubit will be $U_2(U_1|q\rangle)$.

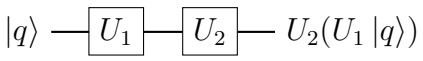

Fig. 4. Two consecutive single-qubit gates.

Exercise 3.4. Show that

$$U_2U_1|q\rangle = \sum_{i,j,k} \alpha_j U_{2,ik}U_{1,kj}|i\rangle\,, \tag{3.16}$$

where U_1 and U_2 are two arbitrary single-qubit gates. Generalize this formula to N consecutive gates.

A set of unitary transformations that play a key role in quantum computation and communication are the *Pauli matrices* (the same Pauli matrices you certainly encountered when you studied the spin of the electron):

$$\sigma_X = X = \begin{bmatrix} 0 & 1 \\ 1 & 0 \end{bmatrix}, \qquad \sigma_Y = Y = \begin{bmatrix} 0 & -i \\ i & 0 \end{bmatrix}, \qquad \sigma_Z = Z = \begin{bmatrix} 1 & 0 \\ 0 & -1 \end{bmatrix} . \tag{3.17}$$

Most of the time we will refer to them as the X, Y, Z *gates* because this is how they are actually called in quantum computing. However, as we will see, the σ notation is sometimes useful.

Among the many properties of the Pauli matrices, I start by reminding you they are Hermitian, $\sigma_a^\dagger = \sigma_a$. In our notation $a = X, Y, Z$. From the physical point of view this is important because it is telling us that the Pauli matrices are observables.

Exercise 3.5. Show that every Pauli matrix σ_a is its own inverse, that is, $(\sigma_a)^2 = I$, where I is the identity matrix. Verify that, however, the product of two different Pauli matrices satisfy $\sigma_a\sigma_b = -\sigma_b\sigma_a$.

Exercise 3.6. Prove that any complex 2×2 matrix can be uniquely written as a linear combination of the Pauli matrices and the identity.

If we apply the Pauli matrices on the computational basis vectors, we get

$$X|0\rangle = \begin{bmatrix} 0 & 1 \\ 1 & 0 \end{bmatrix} \begin{bmatrix} 1 \\ 0 \end{bmatrix} = \begin{bmatrix} 0 \\ 1 \end{bmatrix} = |1\rangle\,, \qquad X|1\rangle = \begin{bmatrix} 0 & 1 \\ 1 & 0 \end{bmatrix} \begin{bmatrix} 0 \\ 1 \end{bmatrix} = \begin{bmatrix} 1 \\ 0 \end{bmatrix} = |0\rangle\,,$$

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

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

This set of relations established by the Pauli matrices between the computational basis vectors, allow us to define the abstract opetators

$$X|0\rangle = |1\rangle\,, \qquad X|1\rangle = |0\rangle\,, \tag{3.18}$$

$$Y|0\rangle = i|1\rangle\,, \qquad Y|1\rangle = -i|0\rangle\,, \tag{3.19}$$

$$Z|0\rangle = |0\rangle\,, \qquad Z|1\rangle = -|1\rangle\,. \tag{3.20}$$

Exercise 3.7. Use the formula (3.14) to check that these operators indeed have the Pauli matrices as representations.

Note that the Pauli operator X flips the computational basis vectors, $X|i\rangle = |\bar{i}\rangle = |1-i\rangle$. So, its action is similar to the classical NOT gate, $\mathrm{NOT}(b) = \bar{b} = 1-b$. This explains why in quantum computing the X operator is called the *bit flit gate* and is usually denoted NOT.

Exercise 3.8. Compute X, Y, Z on $|+\rangle$ and $|-\rangle$. Interpret your results.

Exercise 3.9. What is the geometric interpretation of the action of the Pauli matrices on vectors in the Bloch sphere 1?

In ket-bra notation the Pauli operator X takes the following form,

$$X = |1\rangle\langle 0| + |0\rangle\langle 1|\,. \tag{3.21}$$

Or, in terms of the Hadamard basis vectors,

$$X = |+\rangle\langle +| + |-\rangle\langle -|\,. \tag{3.22}$$

Exercise 3.10. Find the ket-bra expressions for Y and Z.

Exercise 3.11. Using the column vector representation of the computational basis vectors $|0\rangle$ and $|1\rangle$, show that, in fact, the ket-bra expressions above reproduce the Pauli matrices.

Being Hermitian, the Pauli matrices can be used to define the following unitary operators,

$$R_x(\alpha) = e^{-iX\alpha/2}\,, \quad R_y(\beta) = e^{-iY\beta/2}\,, \quad R_z(\gamma) = e^{-iZ\gamma/2}\,. \tag{3.23}$$

where $\alpha, \beta, \gamma \in [0, 2\pi)$. They can be written more compactly as

$$R_a(\theta_a) = e^{-i\sigma_a\theta_a/2}\,. \tag{3.24}$$

The operator $R_a(\theta_a)$ on a single qubit (2.9) acts as a rotation of θ_a radians about the a axis. We can rewrite them using trigonometric functions,

$$R_a(\theta_a) = \cos(\theta_a/2)I - i\sin(\theta_a/2)\sigma_a\,, \tag{3.25}$$

Exercise 3.12. Prove the previous identity.

Exercise 3.13. Suppose that $\hat{\mathbf{n}} = n_x\hat{\mathbf{i}} + n_y\hat{\mathbf{j}} + n_z\hat{\mathbf{k}}$ is a unit normal vector on the Bloch sphere and $\boldsymbol{\sigma} = \sigma_x\hat{\mathbf{i}} + \sigma_y\hat{\mathbf{j}} + \sigma_z\hat{\mathbf{k}}$. Show that a rotation of an angle $\theta_{\hat{\mathbf{n}}}$ about the axis defined by $\hat{\mathbf{n}}$ is given by

$$R_{\hat{\mathbf{n}}}(\theta_{\hat{\mathbf{n}}}) = e^{-i\hat{\mathbf{n}}\cdot\boldsymbol{\sigma}\theta_{\hat{\mathbf{n}}}/2} = \cos(\theta_{\hat{\mathbf{n}}}/2)I - i\sin(\theta_{\hat{\mathbf{n}}}/2)\hat{\mathbf{n}}\cdot\boldsymbol{\sigma}\,. \tag{3.26}$$

Another single-qubit gate which is extensively used in quantum computing is the *Hadamard gate*, defined by its action on the computational basis vectors as follows

$$H|0\rangle = \frac{1}{\sqrt{2}}|0\rangle + \frac{1}{\sqrt{2}}|1\rangle\,, \qquad H|1\rangle = \frac{1}{\sqrt{2}}|0\rangle - \frac{1}{\sqrt{2}}|1\rangle\,, \tag{3.27}$$

that is,

$$H|0\rangle = |+\rangle\,, \qquad H|1\rangle = |-\rangle\,. \tag{3.28}$$

Thus, if a single qubit enters a Hadamard gate, the outgoing state will be

$$H|q\rangle = H(\alpha_0|0\rangle + \alpha_1|1\rangle) = \alpha_0 H|0\rangle + \alpha_1 H|1\rangle = \alpha_0|+\rangle + \alpha_1|-\rangle\,. \tag{3.29}$$

The Hadamard gate, then, takes a state vector in the computational basis and shift it to the Hadamard basis. The converse is also true because

$$H|+\rangle = \frac{1}{\sqrt{2}}H|0\rangle + \frac{1}{\sqrt{2}}H|1\rangle = \frac{1}{\sqrt{2}}|+\rangle + \frac{1}{\sqrt{2}}|-\rangle = |0\rangle\,, \tag{3.30}$$

$$H|-\rangle = \frac{1}{\sqrt{2}}H|0\rangle - \frac{1}{\sqrt{2}}H|1\rangle = \frac{1}{\sqrt{2}}|+\rangle - \frac{1}{\sqrt{2}}|-\rangle = |1\rangle\,, \tag{3.31}$$

so,

$$H|q\rangle_H = H(\alpha_+|+\rangle + \alpha_-|-\rangle) = \alpha_+ H|+\rangle + \alpha_- H|-\rangle = \alpha_+|0\rangle + \alpha_-|1\rangle\,. \tag{3.32}$$

Exercise 3.14. What is the ket-bra expression of the Hadamard gate?

Above we have chosen to introduce the Hadamard gate in terms of its abstract action on the computational basis vectors, however, we could as well have chosen the matrix viewpoint. As you can easily check (do it!), in the computational basis the Hadamard gate has the following matrix representation,

$$H = \frac{1}{\sqrt{2}}\begin{bmatrix} 1 & 1 \\ 1 & -1 \end{bmatrix}\,. \tag{3.33}$$

Exercise 3.15. Compute H^2. How do you interpret this result?

Exercise 3.16. The Pauli and Hadamard gates satisfy the relation $\sigma_a = \pm H\sigma_b H$. Find these relations for all the Pauli gates. Matrices M, such as the Hadamard gate, that satisfy $\sigma_a = \pm M\sigma_b M^\dagger$, are called *Clifford gates*. Show that the Pauli gates are Clifford gates themselves.

So far we have seen the Pauli matrices, rotations and the Hadamard gate. Let us introduce a couple of other useful single-qubit gates.

We know that in quantum mechanics two state vectors that differ by a global phase, actually represent the same quantum system. In the case of a single qubit, we can write this as $|q\rangle \sim e^{i\phi}|q\rangle$. However, if we add a relative phase between the components of a qubit, the two state vectors describe different quantum systems, $\alpha_0|0\rangle + \alpha_1|1\rangle \nsim \alpha_0|0\rangle + e^{i\phi}\alpha_1|1\rangle$. We can add this relative phase factor $e^{i\phi}$ by letting our qubit enter the following gate,

$$P(\phi)|0\rangle = |0\rangle\,, \qquad P(\phi)|1\rangle = e^{i\phi}|1\rangle\,; \tag{3.34}$$

or, in matrix form,

$$P(\phi) = \begin{bmatrix} 1 & 0 \\ 0 & e^{i\phi} \end{bmatrix}\,. \tag{3.35}$$

This unitary is known as the *relative phase gate*. A special case occurs when $\phi = \pi/2$,

$$S|0\rangle = |0\rangle\,, \qquad S|1\rangle = e^{i\pi/2}|1\rangle = i|1\rangle\,. \tag{3.36}$$

This is the *S gate*. Another useful case is when $\phi = \pi/4$,

$$T|0\rangle = |0\rangle\,, \qquad T|1\rangle = e^{i\pi/4}|1\rangle = \frac{1}{\sqrt{2}}(1+i)|1\rangle\,. \tag{3.37}$$

No surprise, this is called the *T gate*, but sometimes it is also called the $\pi/8$ *gate*. In summary, $P(\phi = \pi/2) = S$ and $P(\phi = \pi/4) = T$.

Exercise 3.17. Prove that the S gate is a Clifford gate.

Exercise 3.18. Do you see why the Z gate is also known as the *phase flip gate*?

Exercise 3.19. Why do you think the gate $R = HSH$ is often called the $\sqrt{\text{NOT}}$ gate?

Exercise 3.20. Compute $P^m(\phi)$ for $m = 2, 3, 4, \ldots$. Consider then the cases $\phi = \pi/2$ and $\phi = \pi/4$. How are these P^m's related to the other unitaries?

Exercise 3.21. In general, a relative phase gate is not Hermitian. What condition must a relative phase gate satisfy in order to be Hermitian?

3.3 Multiple Single-Qubit Gates

Before explaining how a general unitary transformation acts on an n qubit, let us first consider the simpler case of a gate that acts independently on the n single qubits of an n-qubit product state,

$$U|q_n\rangle = U_1 \otimes \ldots \otimes U_n\big(|q'\rangle \ldots |q'^n\rangle\big) = U_1|q'\rangle \ldots U_n|q'^n\rangle\,. \tag{3.38}$$

As you see, these special transformations do not produce any entanglement between the single qubits of the incoming product state.

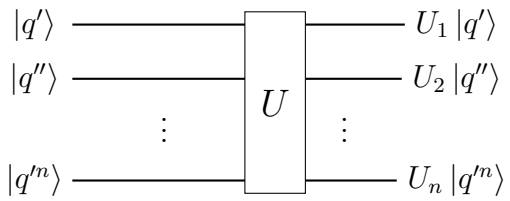

Fig. 5. A non-entangling n-qubit gate.

To be more precise, consider the action of n independent Hadamard gates on the individual qubits of an n-product state,

$$H \otimes \ldots \otimes H\big(|q'\rangle \ldots |q'^n\rangle\big) = H|q'\rangle \ldots H|q'^n\rangle\,. \tag{3.39}$$

Let us start considering the easier cases.

For a computational basis vector $|i\rangle \in \mathcal{H}_q$, a single Hadamard gate acts as follows,

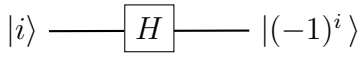

Fig. 6. The Hadamard gate.

Another useful way to write it is

$$H|i\rangle = \frac{1}{\sqrt{2}}|0\rangle + \frac{1}{\sqrt{2}}(-1)^i|1\rangle = \frac{1}{\sqrt{2}}\sum_j (-1)^{ij}|j\rangle\,. \tag{3.40}$$

Since $e^{i\pi} = -1$, a third common notation is

$$H|k\rangle = \frac{1}{\sqrt{2}}|0\rangle + \frac{1}{\sqrt{2}}e^{i\pi k}|1\rangle = \frac{1}{\sqrt{2}}\sum_j e^{i\pi kj}|j\rangle\,. \tag{3.41}$$

Note that in the last equation we used the letter k instead of the usual i to denote the computational basis vectors. We did this simply to avoid confusion with the imaginary i.

Thus, for a single qubit,

$$\begin{aligned} H|q\rangle &= H\sum_i \alpha_i|i\rangle = \sum_i \alpha_i H|i\rangle \\ &= \sum_i \alpha_i|(-1)^i\rangle = \frac{1}{\sqrt{2}}\sum_{i,j}(-1)^{ij}\alpha_i|j\rangle = \frac{1}{\sqrt{2}}\sum_{k,j} e^{i\pi kj}\alpha_k|j\rangle\,. \end{aligned} \tag{3.42}$$

Exercise 3.22. Use the index expressions above to prove that, as we already know from Exercise 3.15, $H(H|i\rangle) = |i\rangle$.

Suppose now we have a product state $|i_1\rangle|i_2\rangle \in \mathcal{H}_{q_2}$ and we apply a Hadamard gate to each of the qubits,

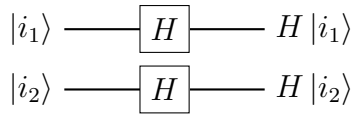

Fig. 7. Two Hadamard gates in parallel.

$$\begin{aligned} H|i_1\rangle H|i_2\rangle &= \frac{1}{\sqrt{2}}\Big(|0\rangle + \frac{1}{\sqrt{2}}(-1)^{i_1}|1\rangle\Big)\frac{1}{\sqrt{2}}\Big(|0\rangle + \frac{1}{\sqrt{2}}(-1)^{i_2}|1\rangle\Big) \\ &= \frac{1}{\sqrt{2^2}}\big(|00\rangle + (-1)^{i_2}|01\rangle + (-1)^{i_1}|10\rangle + (-1)^{i_1+i_2}|11\rangle\big) \\ &= \frac{1}{\sqrt{2^2}}\sum_{j_1,j_2}(-1)^{i_1 j_1 + i_2 j_2}|j_1\rangle|j_2\rangle\,. \end{aligned} \tag{3.43}$$

We can rewrite the left hand side of this equation as follows,

$$H|i_1\rangle H|i_2\rangle = (H\otimes 1)(1\otimes H)|i_1\rangle|i_2\rangle = H\otimes H|i_1\rangle|i_2\rangle = H^{\otimes 2}|i_1\, i_2\rangle\,. \tag{3.44}$$

The right hand side can also be written in a more compact and general form by using the notation $|x\rangle = |i_1\ i_2\rangle$. Similarly, $|y\rangle = |j_1\ j_2\rangle$. Putting these contributions together, we obtain

$$H^{\otimes 2}|x\rangle = \frac{1}{\sqrt{2^2}}\sum_{y}(-1)^{x\cdot y}|y\rangle\,. \tag{3.45}$$

Be aware that here x denotes a binary string and not a decimal number as in equation (2.27). Moreover, $x\cdot y$ is a sort of dot product, $x\cdot y = i_1 j_1 + i_2 j_2$, and not the multiplication of two decimal numbers. Finally, the sum over y simply means

$$\sum_{y} = \sum_{j_1}\sum_{j_2} = \sum_{j_1,j_2}\,. \tag{3.46}$$

You can easily generalize (3.45) to n Hadamard gates acting independently on n single qubits,

$$H^{\otimes n}|x\rangle = \frac{1}{\sqrt{2^n}}\sum_{y}(-1)^{x\cdot y}|y\rangle\,, \tag{3.47}$$

where $x = i_1\ldots i_n$, $y = j_1\ldots j_n$ and $x\cdot y = i_1 j_1 + \ldots + i_n j_n$.

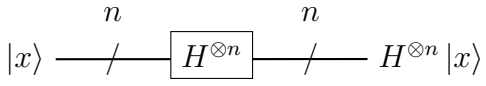

Fig. 8. n Hadamard gates in parallel acting on a computational basis vector of $\mathcal{H}_Q$.

Exercise 3.23. Explain how the general formula (3.47) is obtained by considering three qubits, four qubits, etc.

Exercise 3.24. Show that $H^{\otimes n}(H^{\otimes n}|x\rangle) = |x\rangle$.

If we express the state vector of the n qubit as a linear combination

$$|Q\rangle = \sum_x \alpha_x |x\rangle \,, \tag{3.48}$$

the n Hadamard gates will act according to

$$H^{\otimes n}|Q\rangle = \sum_x \alpha_x H^{\otimes n}|x\rangle = \frac{1}{\sqrt{2^n}} \sum_{x,y} (-1)^{x\cdot y} \alpha_x |y\rangle \,. \tag{3.49}$$

Once again, remember that here, x and y are binary strings.

3.4 Multi-Qubit Gates

So far we have discussed quantum gates that act on single qubits, the natural question now is: what about 2-qubit gates, 3-qubit gates, etc? In principle, nothing prevents us from conceiving quantum gates that act on n qubits. In fact, the mathematical generalization is quite straightforward. If $|Q\rangle$ is the state vector of an n qubit, a general *n-qubit gate* is a unitary transformation U on $|Q\rangle$, $|Q\rangle \mapsto U|Q\rangle$. The only restriction on U is that it must be unitary, $U^{-1} = U^\dagger$.

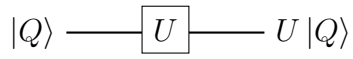

Fig. 9. A unitary transformation acting on an n qubit.

Given that the computational basis vectors of $\mathcal{H}_Q = \mathcal{H}_{q'} \otimes \ldots \otimes \mathcal{H}_{q'^n} \cong \mathbb{C}^{2^n}$ are $|i_1 \ldots i_n\rangle = |i_1\rangle \otimes \ldots \otimes |i_n\rangle$, where $\{|i_r\rangle\} = \{|0\rangle, |1\rangle\}$ is the computational basis of $\mathcal{H}_{q'^r}$, $r = 1, 2, \ldots n$, every vector $|Q\rangle = \sum \alpha_{i_1 \ldots i_n} |i_1 \ldots i_n\rangle$ in $\mathcal{H}_Q$ will have the following column vector representation,

$$|Q\rangle = \sum_{i_1,\ldots,i_n} \alpha_{i_1 \ldots i_n} \begin{bmatrix} \delta_{i_1 0} \\ \delta_{i_1 1} \end{bmatrix} \otimes \ldots \otimes \begin{bmatrix} \delta_{i_n 0} \\ \delta_{i_n 1} \end{bmatrix} \,, \tag{3.50}$$

where $[\delta_{i_r 0} \;\; \delta_{i_r 1}]^T$ is the matrix representation of $|i_r\rangle$.

The unitary transformation U will thus have a $2^n \times 2^n$ matrix representation,

$$U = \begin{bmatrix} U_{11} & \ldots & U_{12^n} \\ \ldots & \ldots & \ldots \\ U_{2^n 1} & \ldots & U_{2^n 2^n} \end{bmatrix} \,. \tag{3.51}$$

For example, the computational basis vectors of $\mathcal{H}_{q_2}$ are simply

$$|0\,0\rangle = \begin{bmatrix} 1 \\ 0 \\ 0 \\ 0 \end{bmatrix} \,, \quad |0\,1\rangle = \begin{bmatrix} 0 \\ 1 \\ 0 \\ 0 \end{bmatrix} \,, \quad |1\,0\rangle = \begin{bmatrix} 0 \\ 0 \\ 1 \\ 0 \end{bmatrix} \,, \quad |1\,1\rangle = \begin{bmatrix} 0 \\ 0 \\ 0 \\ 1 \end{bmatrix} \,. \tag{3.52}$$

It follows that every 2-qubit state vector $|q_2\rangle = \sum \alpha_{ij} |i\,j\rangle$ will be represented by a column vector

$$|q_2\rangle = \alpha_{00} \begin{bmatrix} 1 \\ 0 \\ 0 \\ 0 \end{bmatrix} + \alpha_{01} \begin{bmatrix} 0 \\ 1 \\ 0 \\ 0 \end{bmatrix} + \alpha_{10} \begin{bmatrix} 0 \\ 0 \\ 1 \\ 0 \end{bmatrix} + \alpha_{11} \begin{bmatrix} 0 \\ 0 \\ 0 \\ 1 \end{bmatrix} = \begin{bmatrix} \alpha_{00} \\ \alpha_{01} \\ \alpha_{10} \\ \alpha_{11} \end{bmatrix} \,, \tag{3.53}$$

and every unitary transformation on $|q_2\rangle$ will have the general 4×4 matrix form

$$U = \begin{bmatrix} U_{11} & U_{12} & U_{13} & U_{14} \\ U_{21} & U_{22} & U_{23} & U_{24} \\ U_{31} & U_{32} & U_{33} & U_{34} \\ U_{41} & U_{42} & U_{43} & U_{44} \end{bmatrix} , \tag{3.54}$$

with $U_{rs} = U^*_{sr}$.

Exercise 3.25. Can you find a way to rename the subscripts of the matrix elements U_{rs} of (3.54) so that the product $U|q_2\rangle$ has a tidy form in index notation?

Box 3.1. Tensor product of operators.

In Box 2.1, we recalled the definition of the tensor product of vectors as well as of entire Hilbert spaces. Now, we want to review how operators act on the individual Hilbert spaces of composite systems.

Suppose two single qubits with Hilbert spaces $\mathcal{H}_q$ and $\mathcal{H}_{q'}$ and two operators acting on them,

$$\begin{aligned} A\colon \mathcal{H}_q \to \mathcal{H}_q\,, \qquad & |q\rangle \mapsto A|q\rangle\,, \\ B\colon \mathcal{H}_{q'} \to \mathcal{H}_{q'}\,, \qquad & |q'\rangle \mapsto B|q'\rangle\,. \end{aligned}$$

Let us say we form the 2-qubit system with Hilbert space $\mathcal{H}_{q_2} = \mathcal{H}_q \otimes \mathcal{H}_{q'}$. We can associate to A the operator $A \otimes 1\colon \mathcal{H}_{q_2} \to \mathcal{H}_{q_2}$, such that

$$A \otimes 1|q_2\rangle = A \otimes 1\Big(\sum_{i,j} \alpha_{ij}|i\,j\rangle\Big) = \sum_{i,j} \alpha_{ij}\big(A|i\rangle\big) \otimes 1|j\rangle = \sum_{i,j} \alpha_{ij}\big(A|i\rangle\big)|j\rangle\,.$$

A similar definition applies to the operator B. In general,

$$A \otimes B\Big(\sum_{i,j} \alpha_{ij}|i\,j\rangle\Big) = \sum_{i,j} \alpha_{ij}\big(A|i\rangle\big)\big(B|j\rangle\big)\,. \tag{3.55}$$

Exercise 3.26. Show that a unitary transformation that entangles two single qubits cannot be expressed as the tensor product of two single-qubit gates.

Exercise 3.27. Generalize everything said above for a Hilbert space that is the tensor product of n single qubit spaces.

Given two operators A and B and their respective matrix representations, to the tensor product $A \otimes B$ we associate the matrix

$$A \otimes B = \begin{bmatrix} a_{11} & a_{12} \\ a_{21} & a_{22} \end{bmatrix} \otimes \begin{bmatrix} b_{11} & b_{12} \\ b_{21} & b_{22} \end{bmatrix} = \begin{bmatrix} a_{11}B & a_{12}B \\ a_{21}B & a_{22}B \end{bmatrix} . \tag{3.56}$$

The generalization to more than two operators and to higher order matrices is straightforward.

When there is no risk of confusion, we will drop the tensor product symbol $\otimes$ and simply write AB for $A \otimes B$.

Exercise 3.28. Explain the choice (3.52) for the basis vectors of $\mathcal{H}_{q_2}$.

One of the simplest $2^n \times 2^n$ unitary matrix transformations (3.51) is the one formed by the tensor product of n 2×2 unitary matrices,

$$U = U_1 \otimes \ldots \otimes U_n \,. \tag{3.57}$$

This unitary acts on a product state $|Q\rangle = |q'\rangle \otimes \ldots \otimes |q'^n\rangle$ as follows,

$$U|Q\rangle = U_1 \otimes \ldots \otimes U_n\big(|q'\rangle \otimes \ldots \otimes |q'^n\rangle\big) = U_1\big(|q'\rangle \ldots U_n|q'^n\rangle \,. \tag{3.58}$$

Thus, the unitary (3.57) keeps the quantum state $|Q\rangle$ unentangled. For instance, in the previous subsection we considered $U_1 = \ldots = U_n = H$.

The advantage of a quantum computer over a classical one, though, is its ability to create and efficiently keep track of the superposition of all the possible states available to a quantum system. This includes, of course, entangled states. Thus, if we want to take full advantage of all the power of quantum mechanics, we need to introduce quantum gates that create entanglement. It can be proved that — something we will not do here — a single gate that produces entanglement, in addition to a complete set of single-qubit gates, is all we need to build any multi-qubit gate we want. The gate usually chosen is the so-called CNOT gate. We will first introduce it and then see how it enters into the production of other useful unitaries.

A quantum *controlled gate* is a gate that operates on two qubits, one register by convention called the *control qubit* and the other the *target qubit*. While the control qubit is a single qubit and it remains unchanged when passing through the gate, the target qubit is in general an n qubit and it gets modified depending on the value of the control qubit. By definition, for $c \in \{0, 1\}$, a controlled gate transforms

$$|c\rangle|Q_t\rangle \longmapsto |c\rangle U(c)|Q_t\rangle \,, \tag{3.59}$$

where $U(c)$ is a unitary on $|Q_t\rangle$ which action depends on the value of c.

The *controlled-U gate* is defined as follows,

$$\mathrm{C}U|0\rangle|Q_t\rangle = |0\rangle|Q_t\rangle \,, \qquad \mathrm{C}U|1\rangle|Q_t\rangle = |1\rangle U|Q_t\rangle \,. \tag{3.60}$$

In ket-bra notation,

$$\mathrm{C}U = |0\rangle\langle 0| \otimes 1 + |1\rangle\langle 1| \otimes U \,. \tag{3.61}$$

Its circuit diagram is:

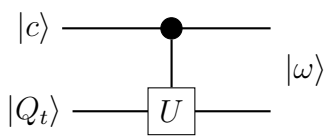

Fig. 10. The controlled-U gate.

Since there is nothing particular about the basis vector $|1\rangle$, we could as well have used the vector $|0\rangle$ to define a controlled gate. The latter is a *controlled-V gate*,

$$\mathrm{C}V|0\rangle|Q_t\rangle = |0\rangle V|Q_t\rangle \,, \qquad \mathrm{C}V|1\rangle|Q_t\rangle = |1\rangle|Q_t\rangle \,. \tag{3.62}$$

As you can show,

$$\mathrm{C}V = |0\rangle\langle 0| \otimes V + |1\rangle\langle 1| \otimes 1 \,. \tag{3.63}$$

The gate is commonly illustrated as follows,

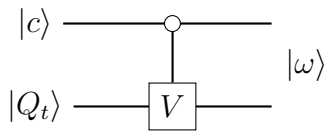

Fig. 11. The controlled-V gate.

We will almost exclusively deal with controlled-U gates.

Note that we can rewrite the definition of a controlled-U gate as follows,

$$\mathrm{C}U|i\rangle|Q_t\rangle = |i\rangle U^i|Q_t\rangle\,, \tag{3.64}$$

where

$$U^i = \begin{cases} 1 \text{ if } i = 0 \\ U \text{ if } i = 1 \end{cases}. \tag{3.65}$$

In particular, if we write the control qubit as $|c\rangle = \sum_i c_i|i\rangle$ and assume that the target qubit is a single qubit with state vector $|t\rangle = \sum_j t_j|j\rangle$, the transformation of a controlled-U gate in index notation takes the general form

$$\mathrm{C}U(|c\rangle|t\rangle) = \sum_{i,j} c_i t_j |i\rangle U^i|j\rangle\,. \tag{3.66}$$

Exercise 3.29. Prove that

$$\mathrm{C}V(|i\rangle|t\rangle) = |i\rangle V^{1-i}|t\rangle\,. \tag{3.67}$$

The matrix representation of a CU gate on single qubits can easily be found:

$$\begin{aligned}
\mathrm{C}U \begin{bmatrix} c_0t_0 \\ c_0t_1 \\ c_1t_0 \\ c_1t_1 \end{bmatrix} &= c_0t_0 \begin{bmatrix} 1 \\ 0 \\ 0 \\ 0 \end{bmatrix} + c_0t_1 \begin{bmatrix} 0 \\ 1 \\ 0 \\ 0 \end{bmatrix} + c_1t_0 \begin{bmatrix} 0 \\ 0 \\ \begin{bmatrix} U & U \\ U & U \end{bmatrix} \begin{bmatrix} 1 \\ 0 \end{bmatrix} \end{bmatrix} + c_1t_1 \begin{bmatrix} 0 \\ 0 \\ \begin{bmatrix} U & U \\ U & U \end{bmatrix} \begin{bmatrix} 0 \\ 1 \end{bmatrix} \end{bmatrix} \\
&= \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \end{bmatrix} \begin{bmatrix} c_0t_0 \\ c_0t_1 \\ 0 \\ 0 \end{bmatrix} + \begin{bmatrix} 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & U_{00} & U_{01} \\ 0 & 0 & U_{10} & U_{11} \end{bmatrix} \begin{bmatrix} 0 \\ 0 \\ c_1t_0 \\ c_1t_1 \end{bmatrix} \\
&= \begin{bmatrix} I & 0 \\ 0 & U \end{bmatrix} \begin{bmatrix} c_0t_0 \\ c_0t_1 \\ c_1t_0 \\ c_1t_1 \end{bmatrix}.
\end{aligned}$$

Thus, we have shown that a controlled-U gate on single qubits has the following matrix representation,

$$\mathrm{C}U = \begin{bmatrix} I & 0 \\ 0 & U \end{bmatrix}. \tag{3.68}$$

Exercise 3.30. What is the matrix representation of a controlled-V gate?

For example, for a *controlled-X gate*,

$$\mathrm{C}X\big(|c\rangle|t\rangle\big) = \mathrm{C}X\Big(\begin{bmatrix}c_0\\c_1\end{bmatrix}\otimes\begin{bmatrix}t_0\\t_1\end{bmatrix}\Big) = \begin{bmatrix}1&0&0&0\\0&1&0&0\\0&0&0&1\\0&0&1&0\end{bmatrix}\begin{bmatrix}c_0t_0\\c_0t_1\\c_1t_0\\c_1t_1\end{bmatrix} = \begin{bmatrix}c_0t_0\\c_0t_1\\c_1t_1\\c_1t_0\end{bmatrix}. \tag{3.69}$$

If the control qubit is in the basis vector $|0\rangle = [1\ 0]^T$, we have

$$\mathrm{C}X\big(|0\rangle|t\rangle\big) = \begin{bmatrix}1&0&0&0\\0&1&0&0\\0&0&0&1\\0&0&1&0\end{bmatrix}\Big(\begin{bmatrix}1\\0\end{bmatrix}\otimes\begin{bmatrix}t_0\\t_1\end{bmatrix}\Big) = \begin{bmatrix}t_0\\t_1\\0\\0\end{bmatrix} = \begin{bmatrix}1\\0\end{bmatrix}\otimes\begin{bmatrix}t_0\\t_1\end{bmatrix} = |0\rangle|t\rangle\,.$$

As expected, since any controlled-U gate does nothing when the control qubit is in the state $|0\rangle$. If, on the other hand, $|c\rangle = |1\rangle = [0\ 1]^T$,

$$\mathrm{C}X(|1\rangle|t\rangle) = \begin{bmatrix}1&0&0&0\\0&1&0&0\\0&0&0&1\\0&0&1&0\end{bmatrix}\Big(\begin{bmatrix}0\\1\end{bmatrix}\otimes\begin{bmatrix}t_0\\t_1\end{bmatrix}\Big) = \begin{bmatrix}0\\0\\t_1\\t_0\end{bmatrix} = \begin{bmatrix}0\\1\end{bmatrix}\otimes\begin{bmatrix}t_1\\t_0\end{bmatrix} = |1\rangle X|t\rangle\,.$$

Exercise 3.31. Show that a *controlled-Z gate* transforms

$$|0\rangle|t\rangle \xmapsto{CZ} |0\rangle|t\rangle\,, \qquad |1\rangle|t\rangle \xmapsto{CZ} |1\rangle\sum_j (-1)^j t_j|j\rangle\,. \tag{3.70}$$

What is the matrix corresponding to CZ?

Exercise 3.32. A useful variant of the relative phase gate (3.34) is the R_l *gate* defined by

$$R_l|j\rangle = e^{\frac{2\pi i}{2^l}j}|j\rangle\,. \tag{3.71}$$

Write its matrix representation. How would you define a controlled-R_l gate? Write the corresponding matrix and draw the circuit diagram.

The *controlled-NOT* or *CNOT gate* is another instance of controlled-U gate on single qubits. For $i, j \in \{0, 1\}$,

$$|i\rangle|j\rangle \xmapsto{\text{CNOT}} |i\rangle|j \oplus i\rangle\,. \tag{3.72}$$

The notation $i \oplus j$ is the standard way of denoting a binary sum: $i \oplus j = (i+j) \bmod 2$. For example, $0 \oplus 0 = 0$, $0 \oplus 1 = 1$, $1 \oplus 0 = 1$ and $1 \oplus 1 = 0$. For the computational basis vectors,

$$\begin{aligned}
|0\rangle|0\rangle &\xmapsto{\text{CNOT}} |0\rangle|0 \oplus 0\rangle = |0\rangle|0\rangle\,,\\
|0\rangle|1\rangle &\xmapsto{\text{CNOT}} |0\rangle|1 \oplus 0\rangle = |0\rangle|1\rangle\,,\\
|1\rangle|0\rangle &\xmapsto{\text{CNOT}} |1\rangle|0 \oplus 1\rangle = |1\rangle|1\rangle\,,\\
|1\rangle|1\rangle &\xmapsto{\text{CNOT}} |1\rangle|1 \oplus 1\rangle = |1\rangle|0\rangle\,.
\end{aligned}$$

That is,

$$|0\,0\rangle \overset{\text{CNOT}}{\longmapsto} |0\,0\rangle\,, \quad |0\,1\rangle \overset{\text{CNOT}}{\longmapsto} |0\,1\rangle\,, \quad |1\,0\rangle \overset{\text{CNOT}}{\longmapsto} |1\,1\rangle\,, \quad |1\,1\rangle \overset{\text{CNOT}}{\longmapsto} |1\,0\rangle\,.$$

From here, we read the matrix representation of the CNOT gate,

$$\text{CNOT} = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 1 & 0 \end{bmatrix} = \text{C}X\,. \tag{3.73}$$

We see that the CNOT gate is actually the same as the CX gate. This is consistent with the fact that the X gate flips the computational basis vectors $|0\rangle \leftrightarrow |1\rangle$ as well as the classical NOT gate flips the bits $0 \leftrightarrow 1$. As we said, the CNOT gate is frequently used to create entanglement and build other unitary transformations.

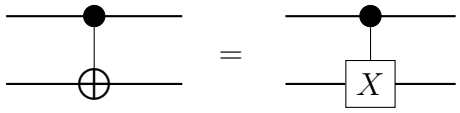

Fig. 12. Circuit identity CNOT=CX.

Exercise 3.33. Show that the CNOT gate is not the tensor product of two single-qubit gates. What is the physical meaning of this?

Exercise 3.34. If $|i\rangle$ is a computational basis state vector, how would you write the transformed state $X|i\rangle$ using the $\oplus$ symbol?

Another useful example of controlled-U gate is the CSWAP gate. In this case the unitary U is a 2-qubit gate known as the *SWAP gate*,

$$|u\rangle|b\rangle \overset{\text{SWAP}}{\longmapsto} |b\rangle|u\rangle\,. \tag{3.74}$$

In components, the SWAP unitary interchanges $u_i \leftrightarrow b_i$. Its matrix representation can be obtained noting that

$$|u\rangle|b\rangle = \begin{bmatrix} u_0b_0 \\ u_0b_1 \\ u_1b_0 \\ u_1b_1 \end{bmatrix} \overset{\text{SWAP}}{\longmapsto} \text{SWAP} \begin{bmatrix} u_0b_0 \\ u_0b_1 \\ u_1b_0 \\ u_1b_1 \end{bmatrix} = \begin{bmatrix} b_0u_0 \\ b_0u_1 \\ b_1u_0 \\ b_1u_1 \end{bmatrix} = \begin{bmatrix} u_0b_0 \\ u_1b_0 \\ u_0b_1 \\ u_1b_1 \end{bmatrix}\,. \tag{3.75}$$

The matrix representation of the SWAP gate in the computational basis of $\mathcal{H}_{q_2}$ is then

$$\text{SWAP} = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}\,. \tag{3.76}$$

Graphically the SWAP gate is represented by the following diagram

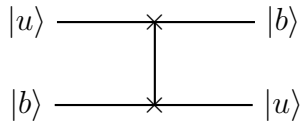

Fig. 13. The SWAP gate.

Exercise 3.35. Show that the ket-bra expression for the SWAP gate is

$$\text{SWAP} = \sum_{k,l} |k\,l\rangle\langle k\,l|\,. \tag{3.77}$$

Exercise 3.36. It is easy to check that the SWAP gate can be written as

$$\text{SWAP} = \frac{1}{2}\big(I\otimes I + X\otimes X + Y\otimes Y + Z\otimes Z\big) = \frac{1}{2}\sum_{A}\sigma_A\otimes\sigma_A\,, \tag{3.78}$$

where $A = I, X, Y, Z$. What is the physical reason for this?

Exercise 3.37. Below is an illustration of the *controlled-SWAP gate* (*CSWAP*). What is the outgoing state $|\omega\rangle$?

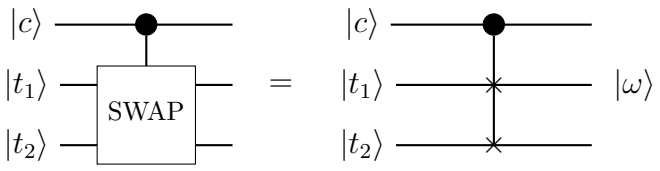

Fig. 14. The controlled-SWAP gate.

Now that we know how to create entangled states from product states using the CNOT gate, we would like to known how to construct other multi-qubit gates using the CNOT gate.

Since quantum gates are identified with unitary transformations, then, according to the mathematical formalism of quantum mechanics, any gate will be the composition of certain unitary transformations (each of them, of course, corresponding to a particular gate),

$$|Q\rangle \mapsto U_1|Q\rangle \mapsto U_2(U_1|Q\rangle) \mapsto \cdots \tag{3.79}$$

In terms of matrices, this means that any quantum gate will be equivalent to a product of matrices, each matrix corresponding to a gate in the circuit. To illustrate how this works, consider the following circuit,

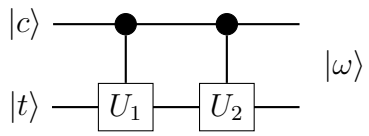

Fig. 15. Two consecutive CU gates.

Exercise 3.38. Verify that each step below is correct:

$$
\begin{aligned}
|c\rangle|t\rangle &= c_0t_0|0\rangle|0\rangle + c_0t_1|0\rangle|1\rangle + c_1t_0|1\rangle|0\rangle + c_1t_1|1\rangle|1\rangle \\
&\xmapsto{CU_1} c_0t_0|0\rangle|0\rangle + c_0t_1|0\rangle|1\rangle + c_1t_0|1\rangle U_1|0\rangle + c_1t_1|1\rangle U_1|1\rangle \\
&\xmapsto{CU_2} c_0t_0|0\rangle|0\rangle + c_0t_1|0\rangle|1\rangle \\
&\quad + c_1t_0|1\rangle\big(U_{1,00}U_2|0\rangle + U_{1,10}U_2|1\rangle\big) + c_1t_1|1\rangle\big(U_{1,01}U_2|0\rangle + U_{1,11}U_2|1\rangle\big) \\
&= c_0t_0|0\rangle|0\rangle + c_0t_1|0\rangle|1\rangle \\
&\quad + \big[c_1t_0\big(U_{1,00}U_{2,00} + U_{1,10}U_{2,01}\big) + c_1t_1\big(U_{1,01}U_{2,00} + U_{1,11}U_{2,01}\big)\big]|1\rangle|0\rangle \\
&\quad + \big[c_1t_0\big(U_{1,00}U_{2,10} + U_{1,10}U_{2,11}\big) + c_1t_1\big(U_{1,01}U_{2,10} + U_{1,11}U_{2,11}\big)\big]|1\rangle|1\rangle\,,
\end{aligned}
$$

which is the product $C\!U_2C\!U_1|c\rangle|t\rangle$.

Exercise 3.39. Compute the evolution of the incoming qubits as they pass through the following gates,

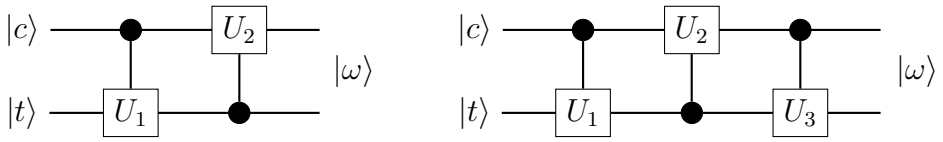

Fig. 16

Exercise 3.40. Show the equivalence of the following circuits,

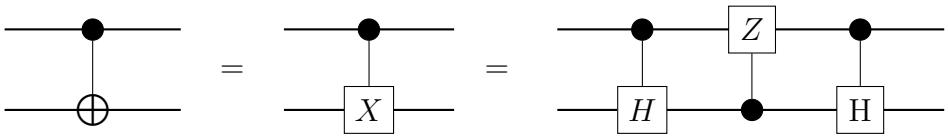

Fig. 17. Circuit identity.

Exercise 3.41. What is the output state $|\omega\rangle$ of the circuit below? What if the CV gate is followed by the CU gate?

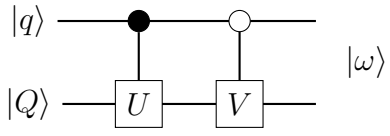

Fig. 18. CU gate followed by a CV gate.

Exercise 3.42. Compare the outgoing states of the following circuits . Then, consider the special case $|t_i\rangle = |0\rangle$ for all $i = 1, \ldots, N$.

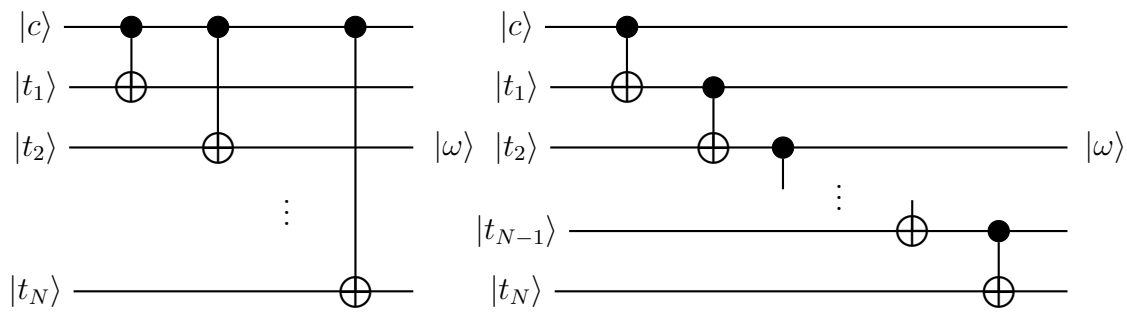

Fig. 19. Circuits (a) and (b).

Exercise 3.43. What sequence of gates undo the action of the gates in Figure 19(b)?

Exercise 3.44. What is the outgoing state of the circuit below? Then, consider the case $U = R_z(\theta)$.

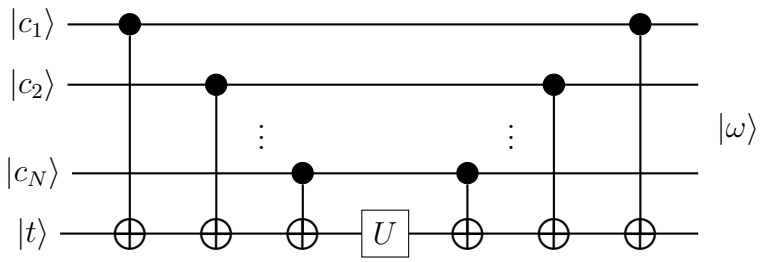

Fig. 20

If, as we said, any multi-qubit gate can be constructed using the CNOT gate and a set of universal single-qubit gates, we should be able to prove that the SWAP and CSWAP gates are concatenations of CNOT gates. The circuit that does it is shown below:

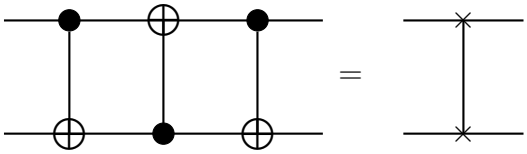

Fig. 21. $\text{CNOT}^3 = \text{SWAP}$.

Let us prove that it does exactly what we want:

$$
\begin{aligned}
|c\rangle|t\rangle = \begin{bmatrix} c_0 \\ c_1 \end{bmatrix} \otimes \begin{bmatrix} t_0 \\ t_1 \end{bmatrix} = \begin{bmatrix} c_0 t_0 \\ c_0 t_1 \\ c_1 t_0 \\ c_1 t_1 \end{bmatrix} \\
&= c_0 t_0 |0\rangle|0\rangle + c_0 t_1 |0\rangle|1\rangle + c_1 t_0 |1\rangle|0\rangle + c_1 t_1 |1\rangle|1\rangle \\
&\xmapsto{\text{CNOT}_2} c_0 t_0 |0\rangle|0\rangle + c_0 t_1 |0\rangle|1\rangle + c_1 t_0 |1\rangle|1\rangle + c_1 t_1 |1\rangle|0\rangle \\
&\xmapsto{\text{CNOT}_1} c_0 t_0 |0\rangle|0\rangle + c_0 t_1 |1\rangle|1\rangle + c_1 t_0 |0\rangle|1\rangle + c_1 t_1 |1\rangle|0\rangle \\
&\xmapsto{\text{CNOT}_2} c_0 t_0 |0\rangle|0\rangle + c_0 t_1 |1\rangle|0\rangle + c_1 t_0 |0\rangle|1\rangle + c_1 t_1 |1\rangle|1\rangle \\
&= \begin{bmatrix} c_0 t_0 \\ c_1 t_0 \\ c_0 t_1 \\ c_1 t_1 \end{bmatrix} = \begin{bmatrix} t_0 \\ t_1 \end{bmatrix} \otimes \begin{bmatrix} c_0 \\ c_1 \end{bmatrix} = |t\rangle|c\rangle \, .
\end{aligned}
$$

Thus, three consecutive CNOT gates acting as shown in Figure 21 are equivalent to a single SWAP gate. We can symbolically write this identity as SWAP = $\text{CNOT}_2\text{CNOT}_1\text{CNOT}_2$. Sometimes this identity is simply written SWAP = CNOT^3.

In index notation the proof is as follows,

$$
\begin{aligned}
|c\rangle|t\rangle = \sum_i c_i |i\rangle \sum_j t_j |j\rangle = \sum_{i,j} c_i t_j |i\rangle|j\rangle \\
&\xmapsto{\text{CNOT}_2} \sum_{i,j} c_i t_j |i\rangle|i \oplus j\rangle \\
&\xmapsto{\text{CNOT}_1} \sum_{i,j} c_i t_j |i \oplus j \oplus i\rangle|i \oplus j\rangle \\
&\xmapsto{\text{CNOT}_2} \sum_{i,j} c_i t_j |i \oplus j \oplus i\rangle|i \oplus j \oplus i \oplus i \oplus j\rangle \, .
\end{aligned}
$$

But, since computational basis vectors satisfy $|i \oplus i\rangle = |0\rangle$, then

$$
|c\rangle|t\rangle \xmapsto{\text{CNOT}_2} \xmapsto{\text{CNOT}_1} \xmapsto{\text{CNOT}_2} \sum_{i,j} c_i t_j |j\rangle|i\rangle = \sum_j t_j |j\rangle \sum_i c_i |i\rangle = |t\rangle|c\rangle \, .
$$

Exercise 3.45. Prove the circuit identity SWAP = CNOT^3 by using the ket-bra form (3.61) of the controlled-U gates.

Exercise 3.46. Show that the CSWAP gate can be implemented by the following equivalent sequence of gates:

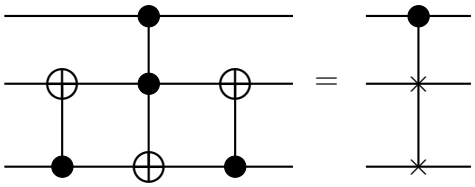

Fig. 22. A sequence of gates equivalent to the CSWAP gate.

In the same spirit, we can construct quantum gates that operate on more than 2 qubits. One such gate is the *controlled-controlled NOT gate*, most commonly known as the *CCNOT* or the *Toffoli gate*. It is a 3-qubit gate that transforms

$$|i\rangle|j\rangle|k\rangle \xmapsto{\text{TOFF}} |i\rangle|j\rangle|k \oplus ij\rangle\,. \tag{3.80}$$

If we write the *first control qubit* as $c_1 = \sum_i c_{1,i}|i\rangle$, the *second control qubit* as $c_2 = \sum_i c_{2,j}|j\rangle$ and the target qubit as $t = \sum_k t_k|k\rangle$, the Toffoli gate transforms

$$|c_1\rangle|c_2\rangle|t\rangle = \sum_{i,j,k} c_{1,i}c_{2,j}t_k|i\rangle|j\rangle|k\rangle \xmapsto{\text{TOFF}} \sum_{i,j,k} c_{1,i}c_{2,j}t_k|i\rangle|j\rangle|k \oplus ij\rangle\,. \tag{3.81}$$

3.5 Measurement

From the beginning of these notes I have assumed that you are familiar with the crucial role played by the measurement process in quantum mechanics. For example, in the pages above I took for granted that you knew that for a single qubit $|q\rangle = \sum_i \alpha_i|i\rangle$, the probability of measuring the state $|i\rangle$ is $|\alpha_i|^2$. Of course, this is simply because

$$P(|i\rangle) = |\langle i|q\rangle|^2 = \Big|\langle i|\sum_j \alpha_j|j\rangle\Big|^2 = \Big|\sum_j \alpha_j\langle i|j\rangle\Big|^2 = \Big|\sum_j \alpha_j\delta_{ij}\Big|^2 = |\alpha_i|^2\,. \tag{3.82}$$

In terms of the projectors on the computational basis vectors, $P_i = |i\rangle\langle i|$, the formula for the probabilities is

$$P(|i\rangle) = |\langle i|q\rangle|^2 = \langle i|q\rangle^*\langle i|q\rangle = \langle q|i\rangle\langle i|q\rangle = \langle q|P_i|q\rangle\,. \tag{3.83}$$

In quantum computing, a measurement in the computational basis of a single qubit is depicted as follows,

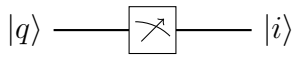

Fig. 23. A measurement gate.

Exercise 3.47. Show that indeed the P_i's are projectors, that is, $P_i^\dagger = P_i$ and $P_i^2 = P_i$.

Exercise 3.48. If a single qubit $|q\rangle$ enters the sequence of gates $HP(\phi)H$, where $P(\phi)$ was defined in (3.34), what is the probability of measuring $|0\rangle$ and $|1\rangle$? Consider then the case $|q\rangle = |0\rangle$. Draw the probabilities for $0 \le \phi \le 2\pi$.

Certainly, there is nothing new here for you. What you may not know, though, is what happens to a 2 qubit when a measurement is performed on only one of the qubits. Let me recall it very quickly.

Given a 2 qubit in a generic state $|q_2\rangle = \sum_{j,k} \alpha_{jk}|jk\rangle$, we can, for example, ask about the probability of finding the first qubit in the computational basis state $|i\rangle$. Because we do nothing to the second qubit, the probability $P(|i\,\cdot\rangle)$ must take into account the two possibilities of the second qubit, that is,

$$\begin{aligned} P(|i\,\cdot\rangle) &= P(|i\,0\rangle) + P(|i\,1\rangle) = |\langle i\,0|q_2\rangle|^2 + |\langle i\,1|q_2\rangle|^2 \\ &= |\alpha_{i0}|^2 + |\alpha_{i1}|^2 = \sum_j |\alpha_{ij}|^2 \,. \end{aligned} \tag{3.84}$$

Similarly, the partial measurement of the second qubit comes with probabilities

$$P(|\,\cdot\,k\rangle) = \sum_i |\alpha_{ik}|^2 \,. \tag{3.85}$$

Exercise 3.49. If $|i\rangle$ is the result of measuring the first qubit, what is the state vector of the second qubit?

Exercise 3.50. Work explicitly the case of 3 qubits. Explore all possible measurements.

To illustrate some interesting consequences of the measurement process in quantum computing, let us consider the following examples. But first, since we will need the controlled-U gate to act on a general target qubit, rather than a single qubit as in (3.66), let us write it again in index notation,

$$|i\rangle|Q\rangle \overset{CU}{\longmapsto} |i\rangle U^i|Q\rangle \,. \tag{3.86}$$

That is, if the incoming product state is $|q\rangle|Q\rangle$, we have that

$$|q\rangle|Q\rangle = \sum_i \alpha_i|i\rangle|Q\rangle \overset{CU}{\longmapsto} \sum_i \alpha_i|i\rangle U^i|Q\rangle \,. \tag{3.87}$$

Without risk of confusion, we can also write this as

$$\sum_i \alpha_i|i\rangle U^i|Q\rangle = \sum_i \alpha_i U^i|i\;Q\rangle \,, \tag{3.88}$$

where it is understood that the U^i in the right hand side is $1 \otimes U^i$.

Let us now examine the following circuit,

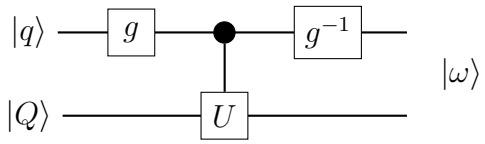

Fig. 24

In this example, g is an arbitrary gate on the single qubit $|q\rangle$ and g^{-1} is its inverse. For instance, g can be any of the Pauli unitaries or the Hadamard gate. The gate

U, on the other hand, is an arbitrary unitary transformation on the n qubit $|Q\rangle$. As a special case, $|Q\rangle$ could be a single qubit.

Following the circuit, we have that

$$|q\,Q\rangle \xmapsto{g} \sum_{i,j} g_{ij}\alpha_i |i\,Q\rangle \xmapsto{CU} \sum_{i,j} g_{ij}\alpha_j U^i |i\,Q\rangle \xmapsto{g^{-1}} \sum_{i,j,k} g^*_{ki} g_{kj} \alpha_j U^k |i\,Q\rangle\,. \tag{3.89}$$

In full form, the output state vector $|\omega\rangle$ is

$$\begin{aligned} |\omega\rangle = {} & \big[\alpha_0(g^*_{00}g_{00} + g^*_{10}g_{10}U) + \alpha_1(g^*_{00}g_{01} + g^*_{10}g_{11}U)\big]|0\,Q\rangle \\ & + \big[\alpha_0(g^*_{01}g_{00} + g^*_{11}g_{10}U) + \alpha_1(g^*_{01}g_{01} + g^*_{11}g_{11}U)\big]|1\,Q\rangle\,. \end{aligned} \tag{3.90}$$

Of course, $g^*_{ij} = g_{ji}$ because g is unitary.

Exercise 3.51. Check that the previous formulas are correct by using explicit matrix representations.

Suppose now that g is the Hadamard gate,

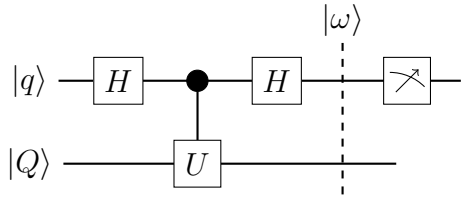

Fig. 25

In this case, the state vector $|\omega\rangle$ is

$$\begin{aligned} |\omega\rangle &= \frac{1}{2}\big[\alpha_0(1+U) + \alpha_1(1-U)\big]|0\,Q\rangle + \frac{1}{2}\big[\alpha_0(1-U) + \alpha_1(1+U)\big]|1\,Q\rangle \\ &= \frac{1}{2}\sum_i \alpha_i\big(1 + (-1)^i U\big)|0\,Q\rangle + \frac{1}{2}\sum_i \alpha_i\big(1 - (-1)^i U\big)|1\,Q\rangle\,. \end{aligned} \tag{3.91}$$

Once again, recall that we are using the shorthand notation AB for the tensor product $A \otimes B$. Thus, by $(1 \pm U)$ we really mean $(1 \otimes 1 \pm 1 \otimes U)$.

The probabilities of measuring the upper qubit in $|0\rangle$ and $|1\rangle$ are

$$P(|0\,\cdot\rangle) = \frac{1}{4}\Big|\sum_i \alpha_i\big(1 + (-1)^i U\big)\Big|^2, \qquad P(|1\,\cdot\rangle) = \frac{1}{4}\Big|\sum_i \alpha_i\big(1 - (-1)^i U\big)\Big|^2.$$

If you prefer, we can write them more compactly as

$$P(|i\,\cdot\rangle) = \frac{1}{4}\Big|\sum_j \alpha_j\big(1 + (-1)^{i+j} U\big)\Big|^2. \tag{3.92}$$

Exercise 3.52. Prove that the sum of these two probabilities is equal to 1.

In particular, if the control qubit $|q\rangle$ in (3.91) is prepared in the state $|0\rangle$, we get

$$|\omega\rangle = \frac{1}{2}(1+U)|0\,Q\rangle + \frac{1}{2}(1-U)|1\,Q\rangle\,. \tag{3.93}$$

Exercise 3.53. Show that

$$P(|0\,\cdot\rangle) = \frac{1}{2}(1+\mathrm{Re}\langle Q|U|Q\rangle)\,. \tag{3.94}$$

Find $P(|1\,\cdot\rangle)$ and show that the sum of the two probabilities is equal to 1.

A similar set-up is at the core of the so called *quantum phase estimation algorithm*. Suppose that the unitary transformation U acts as $U|Q\rangle = e^{i\theta}|Q\rangle$. In other words, assume that the state vector $|Q\rangle$ of the qubit is an eigenvector of the unitary U. In this case, the outgoing state (3.93) becomes

$$|\omega\rangle = \frac{1}{2}(1+e^{i\theta})|0\,Q\rangle + \frac{1}{2}(1-e^{i\theta})|1\,Q\rangle\,. \tag{3.95}$$

As before, we are interested in the probabilities

$$P(|0\,\cdot\rangle) = \frac{1}{4}(1+e^{i\theta})(1+e^{-i\theta}) = \cos^2(\theta/2)\,, \tag{3.96}$$

$$P(|1\,\cdot\rangle) = \frac{1}{4}(1-e^{i\theta})(1-e^{-i\theta}) = \sin^2(\theta/2)\,. \tag{3.97}$$

As you see, there is a closed relationship between these probabilities and the phase angle. For example, if the phase angle is greater that 45°, the probability of measuring the state $|1\rangle$ is greater than measuring $|0\rangle$.

Exercise 3.54. If $U|Q\rangle = e^{i\theta}|Q\rangle$, what is the outgoing state in the following diagram? After this, do it for 3 and — if you can — generalize to an arbitrary number of incoming *measuring qubits* $|0\rangle$.

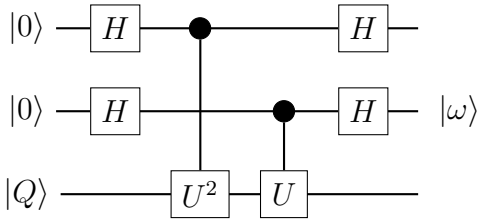

Fig. 26

Another interesting case worth considering is when in the circuit shown in Figure 25, the control qubit is in the state $|q\rangle = (|0\rangle - i|1\rangle)/\sqrt{2}$, for which,

$$|\omega\rangle = \frac{(1-i)}{2\sqrt{2}}(1+iU)|0\,Q\rangle + \frac{(1-i)}{2\sqrt{2}}(1-iU)|1\,Q\rangle\,. \tag{3.98}$$

Exercise 3.55. Draw the circuit diagram that implements the previous transformation. Find the probability $P(|0\,\cdot\rangle)$ and $P(|1\,\cdot\rangle)$.

Suppose now that the incoming qubit $|Q\rangle$ in Figure 25 is a 2-qubit unentangled system, that is, suppose $|Q\rangle = |t_1\rangle|t_2\rangle$, and let U be a SWAP gate. The circuit diagram becomes

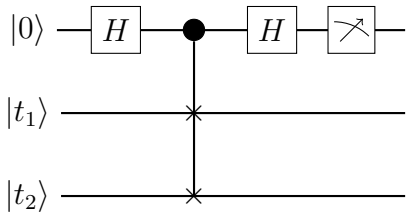

Fig. 27

The analysis of the circuit gives

$$\begin{aligned}
|0\rangle|t_1\rangle|t_2\rangle &\xmapsto{H_c} \frac{1}{\sqrt{2}}\big(|0\rangle + |1\rangle\big)|t_1\ t_2\rangle \\
&\xmapsto{\text{SWAP}} \frac{1}{\sqrt{2}}|0\rangle|t_1\ t_2\rangle + \frac{1}{\sqrt{2}}|1\rangle|t_2\ t_1\rangle \\
&\xmapsto{H_c} \frac{1}{2}\big(|0\rangle + |1\rangle\big)|t_1\ t_2\rangle + \frac{1}{2}\big(|0\rangle - |1\rangle\big)|t_2\ t_1\rangle \\
&= \frac{1}{2}|0\rangle\big(|t_1\ t_2\rangle + |t_2\ t_1\rangle\big) + \frac{1}{2}|1\rangle\big(|t_1\ t_2\rangle - |t_2\ t_1\rangle\big) = |\omega\rangle\,.
\end{aligned}$$

If we measure the first qubit and leave alone the second and third qubits, the probabilities are

$$P(|i\ \cdot\ \cdot\rangle) = P\Big(\frac{1}{2}\big(|i\ t_1\ t_2\rangle + |i\ t_2\ t_1\rangle\big)\Big)\,. \tag{3.99}$$

The 1/2 in the right hand side is the normalization factor.

Exercise 3.56. Prove that, in fact, in the Hilbert space of outgoing states

$$1 \otimes 1 = 1 \otimes |t_1\ t_2\rangle\langle t_1\ t_2| + 1 \otimes |t_2\ t_1\rangle\langle t_2\ t_1|\,. \tag{3.100}$$

The explicit calculation of the probabilities (3.99) is as follows,

$$\begin{aligned}
P(|i\ \cdot\ \cdot\rangle) &= \frac{1}{2}\big(\langle i\ t_1\ t_2| + \langle i\ t_2\ t_1|\big)|\omega\rangle \\
&= \frac{1}{2}\big(\langle i\ t_1\ t_2| + \langle i\ t_2\ t_1|\big)\frac{1}{2}\big(|i\ t_1\ t_2\rangle + (-1)^i|i\ t_2\ t_1\rangle\big) \\
&= \frac{1}{4}\big(\langle t_1|t_1\rangle\langle t_2|t_2\rangle + (-1)^i\langle t_1|t_2\rangle\langle t_2|t_1\rangle \\
&\quad + \langle t_2|t_1\rangle\langle t_1|t_2\rangle + (-1)^i\langle t_2|t_2\rangle\langle t_1|t_1\rangle\big) \\
&= \frac{1}{4}\big[\big(1 + (-1)^i\big) + \big(1 + (-1)^i\big)\langle t_1|t_2\rangle\langle t_1|t_2\rangle^*\big] \\
&= \frac{1}{4}\big(1 + (-1)^i\big)\big(1 + |\langle t_1|t_2\rangle|^2\big)\,.
\end{aligned} \tag{3.101}$$

Exercise 3.57. Show that $P(|0\ \cdot\ \cdot\rangle) + P(|1\ \cdot\ \cdot\rangle) = 1$, regardless of the values of the incoming qubits $|t_1\rangle$ and $|t_2\rangle$.

Notice that, if you prepare the two target qubits $|t_1\rangle$ and $|t_2\rangle$ such that they are perpendicular, $\langle t_1, t_2\rangle = 0$, it follows that $P\big(|0\ \cdot\ \cdot\rangle\big) = 1/2$ and $P\big(|1\ \cdot\ \cdot\rangle\big) = 1/2$ as well. If, instead, they are prepared in the same state, $\langle t_1, t_2\rangle = 1$, the probabilities are $P\big(|0\ \cdot\ \cdot\rangle\big) = 1$ and $P\big(|1\ \cdot\ \cdot\rangle\big) = 0$.

4 Quantum Algorithms

We often hear quantum computing experts and popular science writers alike say that future quantum computers will be much faster than standard computers. They will be so fast that, according to some, in a matter of minutes or even seconds we will be able to solve problems that would take billions of years (more than the age of the universe!) for the most powerful classical supercomputers. Moreover, they say that there is good evidence to think there are problems that, in principle, a quantum computer will be able to solve but classical computers will not, no matter how powerful they become or how much time we give them to work on them. All these claims seem to be unfounded exaggerations, part of the contemporary hype around quantum computers. However, there is something that remains true: there is something in the way a quantum computer processes information — the superposition of quantum states — that has the potential to make it faster than classical computers, at least at solving certain problems.

Note that here we are not referring to the physical realization of these devices, but to the theoretical mode of computation. That is, on paper at least, quantum computers will be faster than classical ones thanks to their unique way of processing information and not because of their implementation. In other words, if we use the quantum circuit model of computation instead of the classical circuit model to design the solution to a problem, we may arrive at a circuit that solves it in less time.

We have been careful to emphasize that quantum computers will be faster than classical ones at solving some problems, but not all. In the circuit model of computation, whether classical or quantum, an *algorithm* is a circuit, that is, a specific arrangements of gates, that given a certain input, delivers the desired output. So, when people, experts and non-experts, loosely say that quantum computers will be much faster than classical computers, what they really mean is that we known some specific quantum algorithms that are faster than the classical algorithms created to solve the same problem.

After all this, you may be wondering, "Ok, but what exactly does "faster" mean?" This is something that, as we will see in the next examples, will depend on each particular problem.

Box 4.1. The probabilistic model of computation.

Another classical model of computation is the probabilistic model. If we have a classical system and there are various outcomes for an experiment, we can use a probabilistic description of the system. For example, when you toss an unbiased coin in the air, you can describe the state of the system with a real two-dimensional vector

$$|C\rangle = \frac{1}{2}|H\rangle + \frac{1}{2}|T\rangle\,. \tag{4.1}$$

The coefficients $1/2$ are the probabilities of observing head or tail when the coin stops. If the coin is biased, the state rector will take the more general form

$$|C\rangle = p_H|H\rangle + p_T|T\rangle\,, \tag{4.2}$$

where $p_H, p_T \in [0,1]$ and $p_H + p_T = 1$. The probabilities p_H and p_T can be obtained by defining an inner product on $\mathbb{R}^2$ such that

$$\langle H|C\rangle = p_H\,, \qquad \langle T|C\rangle = p_T\,. \tag{4.3}$$

No doubt, all this looks very similar to the mathematical description of the single qubit (2.3). We can even write everything in column vectors and introduce matrix transformations on these vectors. It seems that the only difference between the two models is that the coefficients are real in one case and complex in the other. The two descriptions are so similar that many computer scientists prefer to introduce the mathematics of quantum mechanics by using the probabilistic model. Of course, we already knew quantum mechanics, so we did not need to do that.

What we want to stress here, though, is that, despite the resemblance between the mathematics of the probabilistic and the quantum models of computation, there is a fundamental — philosophical, if you wish — difference between the two: the uncertainty in the probabilistic model is due to our limited knowledge of the system, whereas the uncertainty in the quantum model is intrinsic to nature. In principle, we can develop a probabilistic model of computation with complex coefficients (there is nothing wrong with that), but as long as it is based on a physical system which is classical, the superposition of states will not have the same meaning as the superposition of states occurring in the quantum world. In a classical system, whether deterministic or probabilistic, a measurement reveals the true state of the system before the measurement. In contrast, in quantum mechanics, a measurement fixes the state of the system.

4.1 Deutsch's Algorithms

We start with the simplest and historically the first quantum algorithm ever conceived, the algorithm proposed by David Deutsch in 1996 and then we discuss its generalization proposed a few months later by Deutsch himself and Richard Jozsa. The goal of these quantum algorithms is not their real-life application, but to prove that quantum algorithms, at least in principle, can solve computational problems faster than the fastest classical algorithm.

The Deutsch algorithm

Suppose we are given a Boolean function $f\colon \{0,1\} \to \{0,1\}$ and we are told that it is constant or balanced. However, we do not know which of the two is the case. By *constant* we mean that $f(0) = f(1)$, whether because

$$f(0) = f(1) = 0\,, \tag{4.4}$$

or

$$f(0) = f(1) = 1\,, \tag{4.5}$$

On the other hand, *balanced* means that $f(0) \neq f(1)$, that is,

$$f(0) = 0 \neq f(1) = 1\,, \tag{4.6}$$

or

$$f(0) = 1 \neq f(1) = 0\,. \tag{4.7}$$

To keep track of these two possibilities, we will indicate each of the previous cases by f_c and f_b, respectively.

It seems clear that it is not enough to know the value of the function at one single input, whether 0 or 1, to determine if the function is constant or balanced; we need to know the value of the function at both 0 and 1. If the function has to be evaluated at two different values, computer scientists say that the function has to be called "twice" or "two times". In general, the more calls your algorithm makes to a function, the more complex and slow it is. Conversely, the less calls you make to a function, the less complex and faster is your algorithm. This is the principle of what is known as *query complexity*.

What Deutsch discovered is that we can find out whether the function is constant or balanced by calling the function only once. The quantum circuit he conceived was the following,

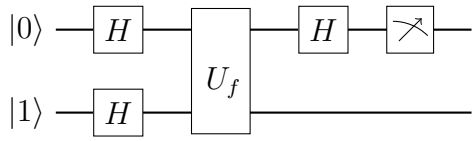

Fig. 28. Circuit configuration for Deutsch's algorithm.

The gate U_f, called an *oracle* or more properly a *XOR oracle*, transforms the computational basis vectors according to

$$|i\rangle|j\rangle \xmapsto{U_f} |i\rangle|j \oplus f(i)\rangle\,. \tag{4.8}$$

Note that it is controlled gate. It is usually depicted as follows,

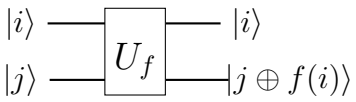

Fig. 29. The oracle of the Deutsch algorithm.

As we said, in the query complexity model we only care about the number of calls made by the algorithm to the function. The inherent complexity proper to the functioning of the oracle is ignored. This is why the oracle is often called a *black box*.

Exercise 4.1. Prove that U_f is unitary.

Exercise 4.2. What is the matrix representation of U_f?

We have all the elements to analyze Deutsch's circuit in Figure 28:

$$\begin{aligned}
|0\,1\rangle &\xmapsto{H\otimes H} |+-\rangle \\
&= \frac{1}{2}\big(|0\,0\rangle - |0\,1\rangle + |1\,0\rangle - |1\,1\rangle\big) \\
&\xmapsto{U_f} \frac{1}{2}U_f\big(|0\,0\rangle - |0\,1\rangle + |1\,0\rangle - |1\,1\rangle\big) \\
&= \frac{1}{2}\big(|0\;0\oplus f(0)\rangle - |0\;1\oplus f(0)\rangle + |1\;0\oplus f(1)\rangle - |1\;1\oplus f(1)\rangle\big)\,. \qquad (4.9)
\end{aligned}$$

Let us first suppose that the function f is constant. Substituting $f(1)$ by $f(0)$ and using f_c instead of f,

$$\begin{aligned}
U_{fc}|+-\rangle &= \frac{1}{2}\big(|0\;0\oplus f_c(0)\rangle - |0\;1\oplus f_c(0)\rangle + |1\;0\oplus f_c(0)\rangle - |1\;1\oplus f_c(0)\rangle\big) \\
&= \frac{1}{2}\big(|0\rangle + |1\rangle\big)|f_c(0)\rangle - \frac{1}{2}\big(|0\rangle + |1\rangle\big)|1\oplus f_c(0)\rangle \\
&= |+\rangle\frac{1}{\sqrt{2}}\big(|f_c(0)\rangle - |1\oplus f_c(0)\rangle\big)\,. \qquad (4.10)
\end{aligned}$$

Consider now the case where f is balanced. Since $f_b(1) = 1 - f_b(0)$, it follows that

$$\begin{aligned}
U_{fb}|+-\rangle &= \frac{1}{2}\big(|0\;f_b(0)\rangle - |0\;1\oplus f_b(0)\rangle + |1\;1-f_b(0)\rangle - |1\;1\oplus 1-f_b(0)\rangle\big) \\
&= \frac{1}{2}|0\rangle\big(|f_b(0)\rangle - |1\oplus f_b(0)\rangle\big) + \frac{1}{2}|1\rangle\big(|1-f_b(0)\rangle - |1\oplus 1-f_b(0)\rangle\big)\,. \qquad (4.11)
\end{aligned}$$

When $f_b(0) = 0$,

$$\begin{aligned}
U_{fb,0}|+-\rangle &= \frac{1}{2}|0\rangle\big(|0\rangle - |1\rangle\big) + \frac{1}{2}|1\rangle\big(|1\rangle - |1\oplus 1\rangle\big) \\
&= \frac{1}{\sqrt{2}}\big(|0\rangle - |1\rangle\big)\frac{1}{\sqrt{2}}\big(|0\rangle - |1\rangle\big) = |--\rangle\,, \qquad (4.12)
\end{aligned}$$

and, when $f_b(0) = 1$,

$$\begin{aligned}
U_{fb,1}|+-\rangle &= \frac{1}{2}|0\rangle\big(|1\rangle - |1\oplus 1\rangle\big) + \frac{1}{2}|1\rangle\big(|0\rangle - |1\oplus 1-1\rangle\big) \\
&= -\frac{1}{\sqrt{2}}\big(|0\rangle - |1\rangle\big)\frac{1}{\sqrt{2}}\big(|0\rangle - |1\rangle\big) = -|--\rangle\,. \qquad (4.13)
\end{aligned}$$

In summary, for f constant,

$$U_{fc}|+-\rangle = \pm|+-\rangle\,, \qquad (4.14)$$

and, for f balanced,

$$U_{fb}|+-\rangle = \pm|--\rangle\,. \qquad (4.15)$$

The last Hadamard gate in Figure 28 gives: for f constant,

$$U_{fc}|+-\rangle \xmapsto{H\otimes 1} \pm(H\otimes 1)|+-\rangle = \pm|0-\rangle\,, \qquad (4.16)$$

and, for f balanced,

$$U_{fb}|+-\rangle \xmapsto{H\otimes 1} \pm(H\otimes 1)|--\rangle = \pm|1-\rangle\,. \tag{4.17}$$

Finally, we measure the state of the upper qubit. If the measurement gives the state $|0\rangle$, then we know with absolute certainty that the function is constant. If, instead, we measure $|1\rangle$, then the function is balanced. This completes the Deutsch algorithm. As stated, we can discover whether the function is constant or balanced by calling it just once.

For completeness' sake, let us present the Deutsch algorithm in a slightly more general form. Suppose that two qubits,

$$|u\rangle = \sum_i u_i|i\rangle\,, \qquad |b\rangle = \sum_j b_j|j\rangle\,, \tag{4.18}$$

enter the oracle in Figure 29. The output is given by,

$$\begin{aligned}|u\rangle|b\rangle = \sum_{i,j} u_i b_j|i\rangle|j\rangle \xmapsto{U_f} \sum_{i,j} u_i b_j|i\rangle|j\oplus f(i)\rangle \\ = u_0b_0|0\rangle|0\oplus f(0)\rangle + u_0b_1|0\rangle|1\oplus f(0)\rangle \\ + u_1b_0|1\rangle|0\oplus f(1)\rangle + u_1b_1|1\rangle|1\oplus f(1)\rangle\,.\end{aligned} \tag{4.19}$$

When the function is constant, $f_c(0) = f_c(1)$, we group the first term with the third and the second with the fourth,

$$U_{fc}(|u\rangle|b\rangle) = \big(u_0b_0|0\rangle + u_1b_0|1\rangle\big)|f_c(0)\rangle + \big(u_0b_1|0\rangle + u_1b_1|1\rangle\big)|1\oplus f_c(0)\rangle\,. \tag{4.20}$$

When f is balanced, $f_b(0) \neq f_b(1)$, we group the first term with the fourth and the second with the third,

$$U_{fb}(|u\rangle|b\rangle) = \big(u_0b_0|0\rangle + u_1b_1|1\rangle\big)|f_b(0)\rangle + \big(u_0b_1|0\rangle + u_1b_0|1\rangle\big)|1\oplus f_b(0)\rangle\,. \tag{4.21}$$

Note that equations (4.20) and (4.21) are telling us that the bottom incoming qubit cannot be in a state with $b_0 = b_1$, if not we would not be able to identify whether f is constant or balanced. So, for the algorithm to work, the first condition is to set $b_0 \neq b_1$. Now, b_0 and b_1 have to be chosen so that a single measurement of the upper qubit will tell us if the function is constant or balanced. In general, of course, $|b_0|^2 + |b_1|^2 = 1$; however, for simplicity we can choose $b_0 = 1/\sqrt{2} = -b_1$. That is, $|b\rangle = |-\rangle$. We then have that

$$\begin{aligned}U_{fc}(|u\rangle|-\rangle) &= \frac{1}{\sqrt{2}}\big(u_0|0\rangle + u_1|1\rangle\big)|f_c(0)\rangle - \frac{1}{\sqrt{2}}\big(u_0|0\rangle + u_1|1\rangle\big)|1\oplus f_c(0)\rangle \\ &= \frac{1}{\sqrt{2}}\big(u_0|0\rangle + u_1|1\rangle\big)\big(|f_c(0)\rangle - |1\oplus f_c(0)\rangle\big)\,,\end{aligned} \tag{4.22}$$

and

$$\begin{aligned}U_{fb}(|u\rangle|-\rangle) &= \frac{1}{\sqrt{2}}\big(u_0|0\rangle - u_1|1\rangle\big)|f_b(0)\rangle - \frac{1}{\sqrt{2}}\big(u_0|0\rangle - u_1|1\rangle\big)|1\oplus f_b(0)\rangle \\ &= \frac{1}{\sqrt{2}}\big(u_0|0\rangle - u_1|1\rangle\big)\big(|f_b(0)\rangle - |1\oplus f_b(0)\rangle\big)\,.\end{aligned} \tag{4.23}$$

Since we want a single measurement on the upper qubit to be able to unambiguously distinguish its state, we need to choose u_0 and u_1 such that the two vectors $(u_0|0\rangle + u_1|1\rangle)/\sqrt{2}$ and $(u_0|0\rangle - u_1|1\rangle)/\sqrt{2}$ are perpendicular. The condition is then,

$$\begin{aligned}\frac{1}{\sqrt{2}}\big(\langle 0|u_0^* + \langle 1|u_1^*\big)\frac{1}{\sqrt{2}}\big(u_0|0\rangle - u_1|1\rangle\big) &= \frac{1}{2}u_0^*u_0 - \frac{1}{2}u_1^*u_1 \\ &= \frac{1}{2}(1 - u_1^*u_1 - u_1^*u_1) = \frac{1}{2} - u_1^*u_1 = 0 \\ &= \frac{1}{2}(u_0^*u_0 - 1 + u_0^*u_0) = u_0^*u_0 - \frac{1}{2} = 0\,.\end{aligned}$$

This is enough to know whether f is constant or balance. For, example, as we did above, the usual choice is $u_0 = u_1 = 1/\sqrt{2}$.

The Deutsch-Jozsa algorithm

Suppose you are given a Boolean function $f\colon \{0,1\}^n \to \{0,1\}$ and you are told that it is constant or balanced. However, you do not know which of the two is the case. Again, as for the Deutsch algorithm (for which $n = 1$), the quantum circuit we will discuss below finds whether the function is constant or balanced by calling the function f a fewer number of times than the classical optimal solution. By *constant* we mean that f takes the same value, 0 or 1, for all the x's in the domain $\{0,1\}^n$. By *balanced* we mean that half of the x's in $\{0,1\}^n$ take the value 0 and the other half the value 1.

Exercise 4.3. Show that these definitions are consistent with the ones given above for the Deutsch algorithm.

To easily generalize to the Boolean function $f\colon \{0,1\}^n \to \{0,1\}$, let us start by considering $n = 1$ and $n = 2$. The case $n = 1$ is just the Deutsch algorithm already discussed. The evolution of the incoming product state $|0\rangle|1\rangle$ as it moves through the circuit shown in Figure 28 is

$$\begin{aligned}|0\,1\rangle &\xmapsto{H\otimes H} H|0\rangle \otimes H|1\rangle = \frac{1}{\sqrt{2}}\sum_i |i\rangle \frac{1}{\sqrt{2}}\sum_k (-1)^k|k\rangle \\ &= \frac{1}{\sqrt{2^2}}\sum_{i,k}(-1)^k|i\ k\rangle \\ &\xmapsto{U_f} \frac{1}{2}\sum_{i,k}(-1)^k|i\ k \oplus f(i)\rangle \\ &\xmapsto{H\otimes 1} \frac{1}{2}\sum_{i,k}(-1)^k H|i\rangle|k \oplus f(i)\rangle \\ &= \frac{1}{2}\sum_{i,k}(-1)^k \frac{1}{\sqrt{2}}\sum_j (-1)^{ij}|j\rangle|k \oplus f(i)\rangle \\ &= \frac{1}{2}\sum_{i,j,k}(-1)^{k+ij}\frac{1}{\sqrt{2}}|j\ k \oplus f(i)\rangle\,. \end{aligned} \tag{4.24}$$

The following is a similar circuit, but with three incoming single qubits instead of two,

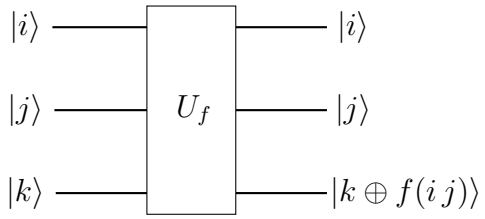

Fig. 30

The oracle in this case transforms

$$|i\rangle|j\rangle|k\rangle \overset{U_f}{\longmapsto} |i\rangle|j\rangle|k \oplus f(i\;j)\rangle\,. \tag{4.25}$$

The incoming state $|0\rangle|0\rangle|1\rangle$ evolves as follows,

$$\begin{aligned}
|0\,0\,1\rangle &\overset{H^{\otimes 2}\otimes H}{\longmapsto} H|0\rangle \otimes H|0\rangle \otimes H|1\rangle = \frac{1}{\sqrt{2}}\sum_{i_1}|i_1\rangle\frac{1}{\sqrt{2}}\sum_{i_2}|i_2\rangle\frac{1}{\sqrt{2}}\sum_{k}(-1)^k|k\rangle \\
&= \frac{1}{\sqrt{2^3}}\sum_{i_1,i_2,k}(-1)^k|i_1\;i_2\;k\rangle \\
&\overset{U_f}{\longmapsto} \frac{1}{2^{3/2}}\sum_{i_1,i_2,k}(-1)^k|i_1\;i_2\;k\oplus f(i_1\;i_2)\rangle \\
&\overset{H^{\otimes 2}\otimes 1}{\longmapsto} \frac{1}{2^{3/2}}\sum_{i_1,i_2,k}(-1)^k H|i_1\rangle H|i_2\rangle|k\oplus f(i_1\;i_2)\rangle \\
&= \frac{1}{2^{3/2}}\sum_{i_1,i_2,k}(-1)^k\frac{1}{\sqrt{2}}\sum_{j_1}(-1)^{i_1 j_1}|j_1\rangle\frac{1}{\sqrt{2}}\sum_{j_2}(-1)^{i_2 j_2}|j_2\rangle|k\oplus f(i_1\;i_2)\rangle \\
&= \frac{1}{2^2}\sum_{i_1,i_2,j_1,j_2,k}(-1)^{k+i_1 j_1+i_2 j_2}\frac{1}{\sqrt{2}}|j_1\;j_2\;\;k\oplus f(i_1\;i_2)\rangle\,.
\end{aligned} \tag{4.26}$$

In general, the control qubits form the state $|0\rangle^{\otimes n}$ and the oracle is

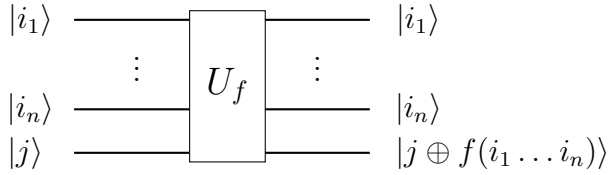

Fig. 31. Oracle of the Deutsch-Jozsa algorithm.

or, in simplified form,

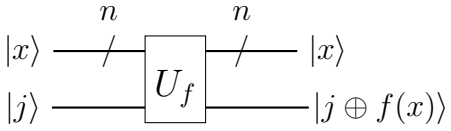

Fig. 32. Simplified diagram for the oracle of the Deutsch-Jozsa algorithm.

By induction, we see that the state of the system right before the measurements is given by,

$$|0\rangle^{\otimes n}\otimes|1\rangle \xmapsto{H^{\otimes n}\otimes H} \xmapsto{U_f} \xmapsto{H^{\otimes n}\otimes 1} |\omega\rangle_{f(x)} = \frac{1}{2^n}\sum_{x,y,k}(-1)^{k+x\cdot y}|y\rangle\frac{1}{\sqrt{2}}|k\oplus f(x)\rangle\,, \tag{4.27}$$

where we are using the dot in $x\cdot y$ to indicate that x and y are in binary notation (not in decimal notation!). Recall the discussion concerning the notation used in equation (3.47). Writing explicitly the sum over k,

$$|\omega_{f(x)}\rangle = \frac{1}{2^n}\sum_{x,y}(-1)^{x\cdot y}|y\rangle\frac{1}{\sqrt{2}}\big(|f(x)\rangle - |1\oplus f(x)\rangle\big)\,. \tag{4.28}$$

Note that $f(x)=0$ gives

$$|\omega\rangle_0 = \frac{1}{2^n}\sum_{x,y}(-1)^{x\cdot y}|y\rangle|-\rangle\,, \tag{4.29}$$

and for $f(x)=1$,

$$|\omega\rangle_1 = \frac{1}{2^n}\sum_{x,y}(-1)^{x\cdot y}|y\rangle(-1)|-\rangle\,. \tag{4.30}$$

Therefore,

$$|0\rangle^{\otimes n}\otimes|1\rangle \longmapsto |\omega\rangle_{f(x)} = \frac{1}{2^n}\sum_{x,y}(-1)^{f(x)+x\cdot y}|y\rangle|-\rangle\,. \tag{4.31}$$

The probability of measuring all the upper qubits in the state $|0\rangle$ is

$$\begin{aligned}
P\big(|0\ldots0\,\cdot\rangle\big) &= \Big|\langle 0\ldots0|\frac{1}{2^n}\sum_{x,y}(-1)^{f(x)+x\cdot y}|y\rangle\Big|^2 \\
&= \frac{1}{2^{2n}}\Big|\sum_{i_1,\ldots,i_n}\sum_{j_1,\ldots,j_n}(-1)^{f(i_1\ldots i_n)+i_1j_1+\ldots+i_1j_n}\langle 0\ldots0|j_1\ldots j_n\rangle\Big|^2 \\
&= \frac{1}{2^{2n}}\Big|\sum_{i_1,\ldots,i_n}\sum_{j_1,\ldots,j_n}(-1)^{f(i_1\ldots i_n)+i_1j_1+\ldots+i_1j_n}\delta_{j_10}\ldots\delta_{j_n0}\Big|^2 \\
&= \frac{1}{2^{2n}}\Big|\sum_x(-1)^{f(x)}\Big|^2\,.
\end{aligned} \tag{4.32}$$

If f is constant, we have two possibilities: whether $f(x)=0$, in which case

$$P\big(|0\ldots0\,\cdot\rangle\big) = \frac{1}{2^{2n}}\big|2^n(-1)^0\big|^2 = \frac{1}{2^{2n}}4^n = 1\,, \tag{4.33}$$

or $f(x)=1$,

$$P\big(|0\ldots0\,\cdot\rangle\big) = \frac{1}{2^{2n}}\big|2^n(-1)^1\big|^2 = \frac{1}{2^{2n}}4^n = 1\,. \tag{4.34}$$

Thus, in both cases the probability of measuring the upper qubits in the state $|0\ldots0\rangle$ is 1.

For f balanced,

$$P\big(|0\ldots0\,\cdot\rangle\big) = \frac{1}{2^{2n}}\Big|\frac{2^n}{2}(-1)^0 + \frac{2^n}{2}(-1)^1\Big|^2 = \frac{1}{2^{2n}}\big|2^{n-1}-2^{n-1}\big|^2 = 0\,. \tag{4.35}$$

What this is saying is that in case the function f is balanced, it is impossible for all the upper qubits to be measured in the state $|0\rangle$; at least one of them is measured in $|1\rangle$.

In conclusion, by just calling once the oracle and choosing the appropriate states to measure, we can determine whether the function f is constant or balanced. In the classical case, best case scenario we had to call the function twice.

Exercise 4.4. Apply the previous analysis to the results (4.24) and (4.26).

4.2 Shor's Factoring Algorithm

Shor's algorithm is without doubt the most famous of all the quantum algorithms conceived so far. When it was invented in the mid-90s, it propelled the field of quantum computing into a new era of development. However, despite its undeniable notoriety, historical importance and possible future application, here we will only give a summary of the concepts it involves and its main attributes. The motivation for this decision is twofold. First, as we said in the introduction, the goal of the present notes is to sketch the main physical ideas and mathematical tools used in quantum computing; alas, Shor's algorithm is too complex to be properly presented in a few pages. Second, Shor's algorithm concerns a rather technical domain of quantum computing, that of secure transfer of information (*cryptography*), and we are instead more interested in the physics of quantum computing.

We start with a rough definition of Shor's algorithm to get an idea of the ingredients involved. Shor's algorithm is a quantum algorithm that solves the problem of finding the prime factors of an integer number faster than any known classical algorithm. The first thing we recognize is that some number theory must be at play here. In addition, the solution found by Shor exploits the connection between prime factoring and something we will describe below as period finding. The latter uses a mathematical technique called the quantum Fourier transform (See Box 4.2).

Simply put, the prime factoring problem asks you to discover the two prime factors of a number that a priori is known to be the product of these numbers. For example, you may be asked to find the prime factors of 15 or 21. Of course, in these simple cases you know that the prime factors are 3,5 and 3,7, respectively. To check it, you simply multiply 3×5 and 3×7. However, it is not so easy to find the prime factors of a larger number such as 755,221. You can check that they are 773 and 977. As you see, if I give you the two prime factors, you can easily verify that they are in fact the correct ones, however, to find them is not so easy. As the number becomes larger and larger, the problem of finding the prime factors becomes harder and harder and eventually impossible to solve by classical computational methods. The difficulty of solving this problem is at the heart of the modern encoding process used to transfer secure data (the RSA cryptosystem). Let us use the examples given above to see how the prime factoring problem translates into the period finding problem and how Shor's algorithm partially solves it.

Let us say we want to find the prime factors of 15. We claim that the following ansatz will give us the prime factors,

$$x^2 = 1 \,\mathrm{mod} 15\,. \tag{4.36}$$

A solution is obviously $x = 4$. However, the equation says much more than that. In

fact, note that

$$\begin{aligned} 4^2 = 1 \,\mathrm{mod} 15 &\Longrightarrow 4^2 - 1 = 0 \,\mathrm{mod} 15 \\ &\Longrightarrow (4+1)(4-1) = 0 \,\mathrm{mod} 15 \\ &\Longrightarrow 5 \cdot 3 = 0 \,\mathrm{mod} 15 \,. \end{aligned}$$

So, the equation $x^2 = 1 \,\mathrm{mod} 15$ actually gives the prime factors of 15. A similar procedure applies to the number 21. We start with

$$x^2 = 1 \,\mathrm{mod} 21 \,, \tag{4.37}$$

and find that

$$\begin{aligned} 8^2 = 1 \,\mathrm{mod} 21 &\Longrightarrow 8^2 - 1 = 0 \,\mathrm{mod} 21 \\ &\Longrightarrow (8+1)(8-1) = 0 \,\mathrm{mod} 21 \\ &\Longrightarrow 3^2 \cdot 7 = 0 \,\mathrm{mod} 21 \,. \end{aligned}$$

By just a slight modification of the previous example, we see that the equation $x^2 = 1 \,\mathrm{mod} 21$ indeed gives the prime factors of 21.

Exercise 4.5. Apply this procedure to find the prime factors of 35.

With the success of these examples at hand, we may be tempted to generalize the formula and say that the two prime factors of any number $N = p_1 p_2$ can be found by solving

$$x^2 = 1 \,\mathrm{mod} N \,. \tag{4.38}$$

As before,

$$\begin{aligned} x^2 = 1 \,\mathrm{mod} N &\Longrightarrow (x+1)(x-1) = 0 \,\mathrm{mod} N \\ &\Longrightarrow p_1^{n_1} p_2^{n_2} = 0 \,\mathrm{mod} N \,. \end{aligned}$$

However, it seems that not all prime factors can be found by using the simple formula (4.38). For example, the method does not apply to the number 77. Instead, we have that

$$\begin{aligned} 9^2 = 4 \,\mathrm{mod} 77 &\Longrightarrow (9+2)(9-2) = 0 \,\mathrm{mod} 77 \\ &\Longrightarrow 11 \cdot 7 = 0 \,\mathrm{mod} 77 \,. \end{aligned}$$

Exercise 4.6. Use this procedure to find the prime factors of 755,221.

Thus, it seems that the problem of finding the prime factors is getting more complicated: not only do we have to find x, but now also the number c in front of mod$N$. Perhaps we should modify the initial ansatz (4.38) as follows,

$$x^2 = c \,\mathrm{mod} N \,. \tag{4.39}$$

However, we do not need to do that. For example, consider again the prime factors of 15. We saw that a solution of (4.36) is

$$2^4 = 1 \,\mathrm{mod} 15 \,, \tag{4.40}$$

but another solution is

$$2^8 = 1 \,\mathrm{mod} 15\,. \tag{4.41}$$

In fact, for any $k = 0, 1, 2, \ldots$, the following are solutions,

$$2^{4k} = 1 \,\mathrm{mod} 15\,, \quad 2^{4k+1} = 2 \,\mathrm{mod} 15\,, \quad 2^{4k+2} = 4 \,\mathrm{mod} 15\,, \quad 2^{4k+3} = 8 \,\mathrm{mod} 15\,. \tag{4.42}$$

Similarly, for 21 all the following are solutions

$$2^{6k} = 1 \,\mathrm{mod} 21\,, \quad 2^{6k+1} = 2 \,\mathrm{mod} 21\,, \quad 2^{6k+2} = 4 \,\mathrm{mod} 21$$

$$2^{6k+3} = 8 \,\mathrm{mod} 21\,, \quad 2^{6k+4} = 16 \,\mathrm{mod} 21\,, \quad 2^{6k+5} = 11 \,\mathrm{mod} 21\,.$$

This periodicity explains why the formula (4.39) is also a solution to the prime factoring problem (in some special cases, of course). Thus, assuming that this procedure applies to the integer number N, its prime factors will be given by the equation

$$a^{rk} = 1 \,\mathrm{mod} N\,, \tag{4.43}$$

or, choosing $k = 1$,

$$a^r = 1 \,\mathrm{mod} N\,. \tag{4.44}$$

The exponent r is called the *period*. Note that the period r is the smallest non-trivial exponent R for which $a^R = 1 \,\mathrm{mod} N$. Since we need to use the difference of squares formula, we have that r must be even,

$$a^r - 1 = 0 \,\mathrm{mod} N \Longrightarrow (a^{r/2} + 1)(a^{r/2} - 1) = 0 \,\mathrm{mod} N\,. \tag{4.45}$$

Exercise 4.7. Show that the method does not apply if $N = 21$ and $a = 5$.

Exercise 4.8. What is the period for $N = 77$ and $a = 3$?

In conclusion, here is the procedure: given a number N, we start by picking an integer a and then we proceed to find the period r. The prime factors of N follows from equation (4.44) (of course, as long as r is even). What Shor's algorithm does is to determine the period r faster than any classical algorithm invented so far.

Exercise 4.9. Show that $a^r = 1 \,\mathrm{mod} N$ is equivalent to

$$1 = a^r \,\mathrm{mod} N\,. \tag{4.46}$$

Our goal then is to show how Shor's algorithm finds the period r of the function

$$f(N, a, r) = a^r \,\mathrm{mod} N\,. \tag{4.47}$$

Here, a is an integer number coprime to N. That is, a is an integer number whose prime factors are not prime factors of N.

As we said, we will not present Shor's algorithm in its most general form. Instead, let us see how it finds the prime factors of 15. The circuit is the following,

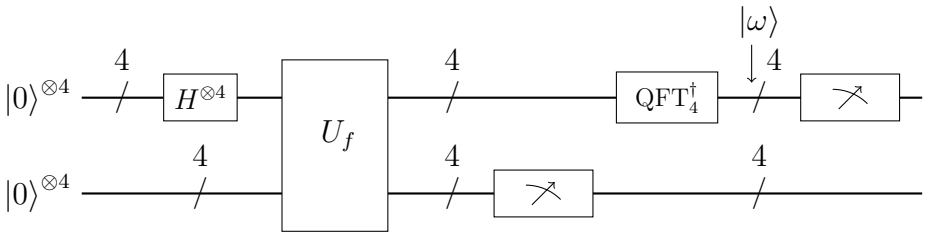

Fig. 33. Circuit designed to find the prime factors of 15.

The oracle U_f is defined by,

$$|x\rangle|0\rangle \xmapsto{U_f} |x\rangle|0 \oplus f(15,13,x)\rangle = |x\rangle|13^x \bmod\rangle\,, \tag{4.48}$$

where everything is written in decimal notation.

The analysis of the circuit is as follows. First, we have the Hadamard gates that act on the four single qubits at the top of the diagram,

$$\begin{aligned}|0\rangle^{\otimes 4}|0\rangle^{\otimes 4} \xmapsto{H^{\otimes 4}\otimes 1^{\otimes 4}} H^{\otimes 4}|0\rangle^{\otimes 4}|0\rangle^{\otimes 4} &= \big(H|0\rangle\big)^{\otimes 4}|0\rangle^{\otimes 4} \\ &= \Big(\frac{1}{\sqrt{2}}\sum_i |i\rangle\Big)^{\otimes 4}|0\rangle^{\otimes 4} = \frac{1}{4}\sum_{i,j,k,l}|i\,j\,k\,l\rangle|0\rangle^{\otimes 4} \\ &= \frac{1}{4}\sum_{x=0}^{15}|x\rangle|0\rangle\,. \end{aligned} \tag{4.49}$$

Be aware that in the last step we changed from binary to decimal notation.

Then, there is the oracle

$$\begin{aligned}&\xmapsto{U_f} \frac{1}{4}\sum_{x=0}^{15}|x\rangle|13^x \bmod 15\rangle \\ &= \frac{1}{4}\big(|0\rangle+|4\rangle+|8\rangle+|12\rangle\big)|1\rangle + \frac{1}{4}\big(|1\rangle+|5\rangle+|9\rangle+|13\rangle\big)|13\rangle \\ &+ \frac{1}{4}\big(|2\rangle+|6\rangle+|10\rangle+|14\rangle\big)|3\rangle + \frac{1}{4}\big(|3\rangle+|7\rangle+|11\rangle+|15\rangle\big)|6\rangle\,. \end{aligned} \tag{4.50}$$

Suppose now that the measurement of the bottom register gives $|6\rangle$. In this case, the state after the measurement is

$$\xmapsto{M} \frac{1}{2}\big(|3\rangle+|7\rangle+|11\rangle+|15\rangle\big)|6\rangle\,. \tag{4.51}$$

If instead of $|6\rangle$, any of the other states came out, ($|1\rangle, |13\rangle$ or $|3\rangle$), the analysis below would be similar.

Then, we have the inverse quantum Fourier transform on the upper register,

$$\xmapsto{\mathrm{QFT}_4^\dagger} \frac{1}{2}\big(\mathrm{QFT}_4^\dagger|3\rangle + \mathrm{QFT}_4^\dagger|7\rangle + \mathrm{QFT}_4^\dagger|11\rangle + \mathrm{QFT}_4^\dagger|15\rangle\big) = |\omega\rangle\,. \tag{4.52}$$

We will sketch how to compute the first of these inverse QFT's, the others are similar. Using the formula (4.61),

$$\begin{aligned}\mathrm{QFT}_4^\dagger|3\rangle &= \frac{1}{4}\sum_{y=0}^{15} e^{\frac{\pi i}{8}3y}|y\rangle \\ &= \frac{1}{4}\big(|0\rangle + e^{\frac{\pi i}{8}3}|1\rangle + e^{\frac{\pi i}{8}6}|2\rangle + e^{\frac{\pi i}{8}9}|3\rangle\big) \\ &+ \frac{1}{4}\big(e^{\frac{\pi i}{8}12}|4\rangle + e^{\frac{\pi i}{8}15}|5\rangle + e^{\frac{\pi i}{8}18}|6\rangle + e^{\frac{\pi i}{8}21}|7\rangle\big) \\ &+ \frac{1}{4}\big(e^{\frac{\pi i}{8}24}|8\rangle + e^{\frac{\pi i}{8}27}|9\rangle + e^{\frac{\pi i}{8}30}|10\rangle + e^{\frac{\pi i}{8}33}|11\rangle\big) \\ &+ \frac{1}{4}\big(e^{\frac{\pi i}{8}36}|12\rangle + e^{\frac{\pi i}{8}39}|13\rangle + e^{\frac{\pi i}{8}42}|14\rangle + e^{\frac{\pi i}{8}45}|15\rangle\big)\,. \end{aligned} \tag{4.53}$$

Computing all of them and substituting in (4.52) yields

$$|\omega\rangle = \frac{1}{2}|0\rangle + \frac{i}{2}|4\rangle - \frac{1}{2}|8\rangle - \frac{i}{2}|12\rangle\,. \tag{4.54}$$

The corresponding probabilities are,

$$P\big(|0\rangle\big) = P\big(|4\rangle\big) = P\big(|8\rangle\big) = P\big(|12\rangle\big) = 1/4\,. \tag{4.55}$$

Exercise 4.10. Do all the calculations that lead to (4.54).

With this, we conclude the quantum analysis of Shor's algorithm. What remains is a classical post-processing, where the period $r = 4$ is found. We will not provide the details here, but you can see it from the possible measurement outcomes $0, 4, 8, 12$.

Box 4.2. The quantum Fourier transform.

One of the mathematical tools used in Shor's algorithm — but also in other quantum algorithms — is the so called *quantum Fourier transform* or *QFT* for short. In few words, the QFT is a change of basis from the computational basis to the Fourier basis.

In (2.27) we saw how we can express any n-qubit state vector $|Q\rangle \in \mathcal{H}_Q \cong \mathbb{C}^{2^n}$ in binary as well as decimal notation,

$$|Q\rangle = \sum_{i_1,\dots,i_n} \alpha_{i_1\dots i_n}|i_1\,\dots\,i_n\rangle = \sum_{x=0}^{N-1} \alpha_x|x\rangle\,, \tag{4.56}$$

where $N = 2^n$. However, the vectors $|x\rangle$ are not the only possible vectors one can use to span $\mathcal{H}_Q$. In fact, an infinite amount of alternative sets can be chosen. One such set is the *Fourier basis* with elements given by the following formula

$$|x\rangle_{\text{QFT}} = \frac{1}{\sqrt{N}} \sum_{y=0}^{N-1} e^{\frac{2\pi i}{N}xy}|y\rangle\,. \tag{4.57}$$

The vector $|x\rangle_{\text{QFT}}$ is the quantum Fourier transformed of $|x\rangle$, that is, $|x\rangle_{\text{QFT}} = \text{QFT}_n|x\rangle$. Be aware that here, y and x must be expressed in decimal notation.

Exercise 4.11. Show that $\text{QFT}_1|i\rangle = |(-1)^i\rangle$, that is, $\text{QFT}_1 = H$.

To show how the quantum Fourier transform works in more complex situations, let us show explicitly how to obtain the Fourier basis vectors of the 2-qubit Hilbert space $\mathcal{H}_{q_2}$, $|j\,k\rangle \mapsto \text{QFT}_2|j\,k\rangle = |j\,k\rangle_{\text{QFT}}$.
The general formula (4.57) in this case is

$$\text{QFT}_2|j\,k\rangle = \text{QFT}_2|x = 2j + k\rangle = \frac{1}{2}\sum_{y=0}^{3} e^{\frac{\pi i}{2}xy}|y\rangle\,. \tag{4.58}$$

The first two Fourier basis vectors are,

$$\begin{aligned}\text{QFT}_2|0\,0\rangle = \text{QFT}_2|x = 0\rangle &= \frac{1}{2}\sum_{y=0}^{3} e^{\frac{\pi i}{2}0y}|y\rangle \\ &= \frac{1}{2}\big(|0\rangle + |1\rangle + |2\rangle + |3\rangle\big) = \frac{1}{2}\big(|0\,0\rangle + |0\,1\rangle + |1\,0\rangle + |1\,1\rangle\big)\,,\end{aligned}$$

and

$$\begin{aligned}\mathrm{QFT}_2|0\,1\rangle = \mathrm{QFT}_2|x=1\rangle &= \frac{1}{2}\sum_{y=0}^{3} e^{\frac{\pi i}{2}1y}|y\rangle \\ &= \frac{1}{2}\big(e^{\frac{\pi i}{2}0}|0\rangle + e^{\frac{\pi i}{2}1}|1\rangle + e^{\frac{\pi i}{2}2}|2\rangle + e^{\frac{\pi i}{2}3}|3\rangle\big) \\ &= \frac{1}{2}\big(|0\,0\rangle + i|0\,1\rangle - |1\,0\rangle - i|1\,1\rangle\big)\,.\end{aligned}$$

Exercise 4.12. Find the Fourier transformed of the other two basis vectors and show that the matrix representation of QFT_2 in the computational basis of $\mathcal{H}_{q_2}$ is

$$\mathrm{QFT}_2 = \frac{1}{2}\begin{bmatrix} 1 & 1 & 1 & 1 \\ 1 & i & -1 & -i \\ 1 & -1 & 1 & -1 \\ 1 & -i & -1 & i \end{bmatrix}\,. \tag{4.59}$$

Exercise 4.13. Find the matrix representation of QFT_3 and QFT_4.

Exercise 4.14. Show that the Fourier basis vectors of $\mathcal{H}_Q$ can also be written

$$\mathrm{QFT}_n|x\rangle = \frac{1}{\sqrt{N}}\bigotimes_{l=1}^{n}\sum_{j} e^{\frac{2\pi i}{2^l}xj}|j\rangle\,. \tag{4.60}$$

Rewrite this expression in terms of $\omega_N = e^{2\pi i/N}$.

Exercise 4.15. What are the matrices of QFT_2, QFT_3 and QFT_4 in terms of $\omega_N = e^{2\pi i/N}$? Do you recognize any pattern? What about QFT_n?

If the QFT takes us from a description of the qubit state vector in the computational basis to the Fourier basis, the *inverse QFT*, denoted $\mathrm{QFT}_n^\dagger = \mathrm{QFT}_n^{-1}$, does precisely the opposite,

$$\mathrm{QFT}_n^\dagger|x\rangle_{\mathrm{QFT}} = \frac{1}{\sqrt{N}}\sum_{y=0}^{N-1} e^{-\frac{2\pi i}{N}xy}|y\rangle\,. \tag{4.61}$$

4.3 Superdense Coding and Teleportation

Recall that, by definition, a separable or product state can always be written as the tensor product of two state vectors. In contrast, an entangled state is non-separable (see equation (2.24)). By abuse of language, even though a 2 qubit is generally entangled and its state vector is not the product of two single-qubit state vectors, in the literature the vectors of the first and second Hilbert spaces are frequently called "first" and "second" qubits, respectively.

Suppose now the following situation,

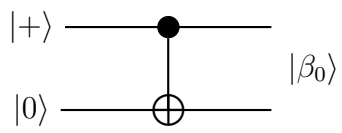

Fig. 34. Production of a Bell state using the CNOT gate.

Initially, the two single-qubit states $|+\rangle$ and $|0\rangle$ are not entangled, so the composite system is in the product state $|+\rangle|0\rangle$. After the CNOT gate is applied, the outgoing state is

$$|+\rangle|0\rangle = \frac{1}{\sqrt{2}}\big(|0\,0\rangle + |1\,0\rangle\big) \xmapsto{\text{CNOT}} \frac{1}{\sqrt{2}}\big(|0\,0\rangle + |1\,1\rangle\big) \equiv \beta_0\,.$$

This is an entangled state in $\mathcal{H}_{q_2}$ and it is called a Bell state. Let us momentarily denote it by β_0. An alternative, but obviously equivalent form of creating β_0 is illustrated in the following diagram,

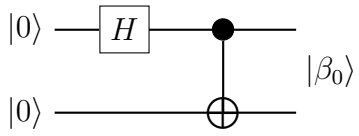

Fig. 35. Alternative set-up to the circuit in Figure 34.

More generally, we can allow the incoming states to be in any of the computational basis state vectors,

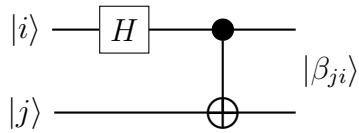

Fig. 36. Set-up to create any Bell state.

The possibilities are:

$$\begin{aligned}
|0\rangle|0\rangle &\xmapsto{H\otimes 1} \frac{1}{\sqrt{2}}\big(|0\rangle + |1\rangle\big)|0\rangle \xmapsto{\text{CNOT}} \frac{1}{\sqrt{2}}\big(|0\,0\rangle + |1\,1\rangle\big) \equiv |\beta_0\rangle\,,\\
|0\rangle|1\rangle &\xmapsto{H\otimes 1} \frac{1}{\sqrt{2}}\big(|0\rangle + |1\rangle\big)|1\rangle \xmapsto{\text{CNOT}} \frac{1}{\sqrt{2}}\big(|0\,1\rangle + |1\,0\rangle\big) \equiv |\beta_1\rangle\,,\\
|1\rangle|0\rangle &\xmapsto{H\otimes 1} \frac{1}{\sqrt{2}}\big(|0\rangle - |1\rangle\big)|1\rangle \xmapsto{\text{CNOT}} \frac{1}{\sqrt{2}}\big(|0\,0\rangle - |1\,1\rangle\big) \equiv |\beta_2\rangle\,,\\
|1\rangle|1\rangle &\xmapsto{H\otimes 1} \frac{1}{\sqrt{2}}\big(|0\rangle - |1\rangle\big)|1\rangle \xmapsto{\text{CNOT}} \frac{1}{\sqrt{2}}\big(|0\,1\rangle - |1\,0\rangle\big) \equiv |\beta_3\rangle\,.
\end{aligned}$$

The four states $|\beta_0\rangle, |\beta_1\rangle, |\beta_2\rangle, |\beta_3\rangle \in \mathcal{H}_{q_2}$ are called *Bell states* or *EPR pairs*. Note that they are perpendicular, so they form a basis for $\mathcal{H}_{q_2}$. This basis is called the *Bell basis*. Of course, we could also have obtained them in a quicker way by using

index notation,

$$|i\rangle|j\rangle \xmapsto{H\otimes 1} H|i\rangle|j\rangle = \frac{1}{\sqrt{2}}\big(|0\rangle + (-1)^i|1\rangle\big)|j\rangle \tag{4.62}$$

$$\xmapsto{\text{CNOT}} \frac{1}{\sqrt{2}}\big(|0\rangle|j\rangle + (-1)^i|1\rangle|\bar{j}\rangle\big) = |\beta_{ji}\rangle\,. \tag{4.63}$$

Interchanging $i \leftrightarrow j$, the general Bell state vector becomes

$$|\beta_{ij}\rangle = \frac{1}{\sqrt{2}}\big(|0\,i\rangle + (-1)^j|1\,\bar{i}\rangle\big)\,. \tag{4.64}$$

Comparing with the states $|\beta_i\rangle$ defined above, we see that $|\beta_{00}\rangle = |\beta_0\rangle$, $|\beta_{01}\rangle = |\beta_1\rangle$, $|\beta_{10}\rangle = |\beta_2\rangle$ and $|\beta_{11}\rangle = |\beta_3\rangle$.

Exercise 4.16. Write the four computational basis vectors of $\mathcal{H}_{q_2}$ in terms of the Bell states.

Superdense coding is a quantum communication protocol designed to communicate two classical bits of information ($b_1b_2 = 00, 01, 10$ or 11) by sending only one single qubit. That is, the code can be used to communicate one of four classical pieces of information: it can be four numbers, four colors, etc. It works as follows. Imagine that there is a sender and a receiver, each with a physical qubit of a 2-qubit system forming a Bell state. The following sequence of unitary transformations shows how the protocol operates.

At the sender's side:

$$\begin{aligned}|\beta_{00}\rangle = \frac{1}{\sqrt{2}}\big(|0\,0\rangle + |1\,1\rangle\big) &\longmapsto \frac{1}{\sqrt{2}}\big(|0\oplus b_2\;\,0\rangle + |1\oplus b_2\;\,1\rangle\big)\\ &\longmapsto \frac{1}{\sqrt{2}}\big((-1)^{b_1b_2}|b_2\;\,0\rangle + (-1)^{b_1(b_2\oplus 1)}|1\oplus b_2\;\,1\rangle\big) = |q\rangle\,.\end{aligned}$$

At the receiver's side:

$$\begin{aligned}|q\rangle &\longmapsto \frac{1}{\sqrt{2}}\big((-1)^{b_1b_2}|b_2\;\,0\oplus b_2\rangle + (-1)^{b_1\bar{b_2}}|1\oplus b_2\;\,1\oplus(1\oplus b_2)\rangle\big)\\ &= |(-1)^{b_1}\rangle|b_2\rangle \qquad \text{(up to a phase)}\\ &\longmapsto |b_1\rangle|b_2\rangle\\ &\longmapsto b_1b_2\,.\end{aligned}$$

Exercise 4.17. First, determine each of transformations indicated above by arrows, then draw the circuit.

Exercise 4.18. Repeat the previous steps in case the preshared 2 qubit is a general Bell state $|\beta_{ij}\rangle$.

Quantum teleportation is a communication protocol designed to transfer the information of a single qubit through a classical channel. The circuit diagram is similar to that of superdense coding, the difference being that the parts of the sender and the receiver are interchanged.

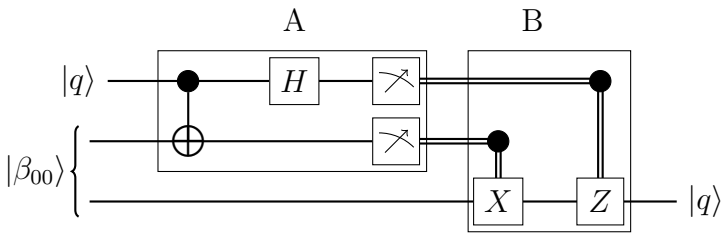

Fig. 37. The quantum teleportation circuit.

Remember that the transfer of classical bits is represented graphically by a double line, while we use single lines for qubits.

Exercise 4.19. Write down the evolution of the initial state $|q\rangle|\beta_{00}\rangle$ at every step of the circuit and show that the outgoing qubit is indeed $|q\rangle$.

Exercise 4.20. Compare the circuit for quantum teleportation shown in Figure 37 with the circuit you drew in Exercise 4.17 for superdense coding.

4.4 Quantum Simulation

The simulation of a quantum mechanical system by using a quantum computer was Feynman's seminal idea on quantum computers. He was convinced that quantum systems, such as common molecules, were so complex that the only way to predict their behaviour was through the use of a device fully built according to the same physical principles as the system itself. Despite Feynman's early vision and the effort made for more than twenty years in that direction, quantum simulation remains a challenging problem. It is not difficult to see why this is so.

Suppose, for example, that you have a quantum system of n interacting particles (let us say electrons), each with two possible quantum states (the electrons can be up or down). A fully quantum mechanical description of the system should keep track of the quantum superposition of the 2^n possible configurations of the system at every time t. If the number of particles is small and the interactions are simple enough, we may expect a classical computer to do the job. However, as soon as the number of particles increases substantially, for instance to $n = 100$, the number of possible configurations to keep track of becomes so large that the problem becomes intractable for classical computers. For this, we need quantum computers.

As you know very well, the dynamics of a quantum system is described by the *Schrödinger equation*,

$$\hat{H}|\psi(t)\rangle = i\frac{d}{dt}|\psi(t)\rangle\,, \tag{4.65}$$

where $\hat{H} = \hat{H}^\dagger$ is the *Hamiltonian operator* and $|\psi(t)\rangle$ is the state of the system at some time t. That is, if $|\psi(t_0)\rangle$ is the state of the system at time t_0, the Schrödinger equation tells you that, for time-independent Hamiltonians, there is an operator

$$U(t, t_0) = e^{-i\hat{H}(t-t_0)}\,, \tag{4.66}$$

called the *time evolution operator*, such that the initial state evolves in time according to

$$|\psi(t_0)\rangle \mapsto |\psi(t)\rangle = U(t, t_0)|\psi(t_0)\rangle\,. \tag{4.67}$$

Exercise 4.21. Show that $|\psi(t)\rangle = e^{-i\hat{H}(t-t_0)}|\psi(t_0)\rangle$ is, indeed, a solution to the Schrödinger equation (4.65).

The idea of *quantum simulation*, also known as *Hamiltonian simulation*, consists of finding a quantum circuit (built, of course, from elementary gates) matching as accurately as possible the time-evolution operator of the real physical system. Here we will only discuss the simulation of the time-evolution operator and assume that we know how to create an n-qubit state $|Q\rangle$ that reproduces the initial state vector $|\psi(t_0)\rangle$ of the system, $|\psi(t_0)\rangle = |Q\rangle$.

To start with, suppose the simplest case of a two-level quantum system with Hamiltonian $\hat{H} = \hat{H}_{q_1}$. The matrix representation of the Hamiltonian in the computational basis is

$$\hat{H}_{q_1} = \begin{bmatrix} H_{00} & H_{01} \\ H_{10} & H_{11} \end{bmatrix} ; \tag{4.68}$$

where, because $\hat{H}$ is Hermitian, $H_{01} = H_{10}^*$. Now, since we know that any complex 2×2 matrix can be written as a linear combination of the Pauli matrices and the identity matrix, then

$$\hat{H}_{q_1} = \sum_A h_A \, \sigma_A \, , \tag{4.69}$$

where $A = I, X, Y, Z$ and, because of the hermicity of the Hamiltonian, $h_A \in \mathbb{R}$.

For the Hamiltonian of a 2-qubit quantum system, we can use an analogous result stating that any Hermitian 4×4 complex matrix can be written as a real linear combination of the tensor product of Pauli matrices,

$$\hat{H}_{q_2} = \sum_{A,B} h_{AB} \, \sigma_A \otimes \sigma_B \, . \tag{4.70}$$

Exercise 4.22. Write the matrices $\hat{H}_{q_1}$ and $\hat{H}_{q_2}$ as a linear combination of the Pauli matrices, displaying explicitly the coefficients h_A and h_{AB}.

Similarly, the most general Hamiltonian for a physical system corresponding to an n qubit is of the form

$$\hat{H}_{q_n} = \sum_{A_1,\ldots,A_n} h_{A_1 \ldots A_n} \, \sigma_{A_1} \otimes \ldots \otimes \sigma_{A_n} \, , \tag{4.71}$$

where $A_1, \ldots, A_n = I, X, Y, Z$ and all the coefficients $h_{A_1 \ldots A_n}$ are real.

Let us now see some simple examples. Suppose we know that the Hamiltonian of a two-level system has the form of the Pauli operator Z, that is, $\hat{H} = \hat{H}_{q_1} = Z$. If $|q\rangle$ is associated to the initial state $|\psi(t_0)\rangle$, the evolution of the physical system will be described by $|q\rangle \mapsto U(t)|q\rangle = e^{-iZt}|q\rangle$. We now recall that the elementary gate $R_z(\theta) = e^{-iZ\theta/2}$, which implies that

$$U(t) = e^{-iZt} = R_z(2t) \, . \tag{4.72}$$

The quantum circuit that simulates the evolution of our physical system is then

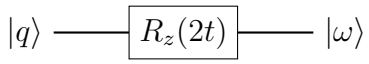

Fig. 38

The state $|\omega\rangle$ leaving the gate is assumed to perfectly match the final state $|\psi(t)\rangle$ of the real physical system we wanted to simulate.

Exercise 4.23. What if the Hamiltonian of the system is any of the other Pauli operators? For instance, for $\hat{H} = X$, show that the quantum circuit modelling the time-evolution operator is

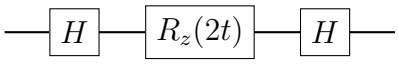

Fig. 39

To find a quantum circuit that simulates the evolution a 2 qubit is more difficult. Suppose, for simplicity, that $\hat{H} = \hat{H}_{q_2} = \sigma_A \otimes \sigma_A$, where $A = I, X, Y, Z$. The time-evolution operator is then $U(t) = e^{-i\sigma_A \otimes \sigma_A t}$. By Taylor expansion,

$$U(t) = e^{-i\sigma_A \otimes \sigma_A t} = \cos(t) I - i \sin(t) \sigma_A \otimes \sigma_A \,. \tag{4.73}$$

Exercise 4.24. Prove the previous formula.

Exercise 4.25. What is the matrix representation of the operator $U(t) = e^{-i\sigma_A \otimes \sigma_A t}$?

If, as we are assuming, $|\psi(t_0)\rangle = |q\rangle$, then the time evolution of the system will be given by,

$$\begin{aligned} U(t)|q_2\rangle &= e^{-i\sigma_A \otimes \sigma_A t} \sum_{i,j} \alpha_{ij} |ij\rangle \\ &= \cos(t) \sum_{i,j} \alpha_{ij} |ij\rangle - i \sin(t) \sum_{i,j} \alpha_{ij} \sigma_A |i\rangle \sigma_A |j\rangle \,. \end{aligned} \tag{4.74}$$

Exercise 4.26. Show that the time evolution operator $U(t) = e^{-iZ \otimes Zt}$ can be implemented by the circuit

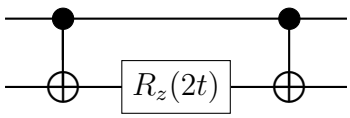

Fig. 40. Circuit emulating $U(t) = e^{-iZ \otimes Zt}$.

Suppose now, more generally, a physical system modelled by an n qubit evolving with $U(t) = e^{-i\sigma_A^{\otimes n}} t$. Taylor expanding as before, we get that

$$U(t) = e^{-i\sigma_A^{\otimes n}} = \cos(t) I - i \sin(t) \sigma_A^{\otimes n} \,. \tag{4.75}$$

Exercise 4.27. What is the quantum circuit corresponding to $U(t) = e^{-iZ^{\otimes n}}$? Compare your diagram with the circuit shown in Exercise 3.44.

Box 4.3. The Pauli group.

As you can easily verify, the product of two Pauli matrices is, up to a constant that can be ± 1 or $\pm i$, another Pauli matrix.

Exercise 4.28. Prove that the Pauli matrices form a group.

The Pauli matrices and the 2×2 identity matrix are said to form the single-qubit Pauli group $\mathcal{P}_1$. To remember that there are constants ± 1 and $\pm i$ involved, the group is commonly denoted

$$\mathcal{P}_1 = \{I, X, Y, Z; \pm 1, \pm i\} \,. \tag{4.76}$$

We can be more economical and write $\mathcal{P}_1 = \{\sigma_A; \pm 1, \pm i\}$, where it is understood that $A = I, X, Y, Z$.

The 2-qubit Pauli group $\mathcal{P}_2$, which acts on 2 qubits, is

$$\mathcal{P}_2 = \{\sigma_A \otimes \sigma_B; \pm 1, \pm i\} \,, \tag{4.77}$$

where $A, B = I, X, Y, Z$. An arbitrary element of $\mathcal{P}_2$ acts as follows,

$$(\sigma_A \otimes \sigma_B)|q_2\rangle = (\sigma_A \otimes \sigma_B)\Big(\sum_{i,j} \alpha_{ij}|i\rangle \otimes |j\rangle\Big) = \sum_{i,j} \alpha_{ij}\, \sigma_A|i\rangle \otimes \sigma_B|j\rangle \tag{4.78}$$

In general, the Pauli group on n qubits, also known as the n-qubit Pauli group for short, is denoted

$$\mathcal{P}_n = \{\sigma_{A_1} \otimes \ldots \otimes \sigma_{A_n}; \pm 1, \pm i\} \,. \tag{4.79}$$

The elements of a Pauli group are called *Pauli operators*. Sometimes we simply write $\sigma_{A_1} \otimes \ldots \otimes \sigma_{A_n} = G_n$. The following alternative notations can be used to denote a Pauli group,

$$\mathcal{P}_n = \{\sigma_A; \pm 1, \pm i\}^{\otimes n} = \{G_n; \pm 1, \pm i\} = {\mathcal{P}_1}^{\otimes n} \,. \tag{4.80}$$

It can be shown that, in general, any $2^n \times 2^n$ complex matrix M can be written as a linear combination

$$M = \sum_{r=1}^{4^n} a_r G_r \,. \tag{4.81}$$

Exercise 4.29. Show that any 4×4 complex matrix is a linear combination of tensor products $\sigma_A \otimes \sigma_B$.

For example, for the Hamiltonian of an n qubit, instead of (4.71), we can write

$$\hat{H} = \sum_{r=1}^{4^n} h_r G_r \,, \tag{4.82}$$

where the hermicity of the Hamiltonian implies that $h_r^\dagger = h_r$.

In the examples above, we have described the Hamiltonians of systems with very simple and somehow unrealistic behaviours. To tackle more interesting situations, we need more powerful methods. One of the simplest approaches is the so called *product formula simulation*.

First, we start by writing the total Hamiltonian as a sum of operators,

$$\hat{H} = \sum_{l=1}^{L} \hat{H}_l \,, \tag{4.83}$$

where each $\hat{H}_l$ acts, at most, on k qubits. These individual terms are called *k-local Hamiltonians*. We now divide the total time interval t into N subintervals, $\Delta t = t/N$ (for simplicity, we are taking $t_0 = 0$), and then use the *product formula*

$$e^{-i(A+B)t} = e^{-iAt} e^{-iBt} + O(\|[A,B]\| t^2/N) \,, \tag{4.84}$$

to finally obtain

$$\begin{aligned} U(t) &= \lim_{N\to\infty} \big(U(t/N)\big)^N \\ &\approx \big(U(t/N)\big)^N = \big(e^{-i\hat{H}t/N}\big)^N = \big(e^{-i\sum_{l=1}^{L}\hat{H}_l t/N}\big)^N \\ &= \big(e^{-i\hat{H}_1 t/N} \dots e^{-i\hat{H}_L t/N}\big)^N + O\Big(\sum_{l_1,l_2}^{L} \|[\hat{H}_{l_1}, \hat{H}_{l_2}]\| t^2/N\Big) \,. \end{aligned} \tag{4.85}$$

Exercise 4.30. Prove the product formula (4.84).

If we have enough reasons to neglect the higher-order terms in (4.85), the evolution of the initial state of the physical system will be given by

$$|\psi(t)\rangle \approx \left(\prod_{l=1}^{L} e^{-i\hat{H}_l t/N}\right)^N |\psi(t_0)\rangle \,. \tag{4.86}$$

Thanks to this approximation, we do not need to find a quantum circuit for the entire time-evolution operator $U(t)$, but for the more manageable short-time operators

$$U_l(t/N) = e^{-i\hat{H}_l t/N} \,. \tag{4.87}$$

Depending on the accuracy of the approximation, we then expect

$$|\psi(t)\rangle \approx \left(\prod_{l=1}^{L} U_l(t/N)\right)^N |q_n\rangle \,. \tag{4.88}$$

5 Quantum Error Correction

Quantum computers are fragile objects, notably because their interactions with the environment, for example, with external electromagnetic fields or tiny temperature changes, produce undesirable perturbations that put at risk the performance of the device and ultimately our confidence in the computation. Knowing that these perturbations are unavoidable, from the very early days of quantum computer science,

experts have been trying to build a theory to understand and have control over them.

The interaction of a quantum system, in our case a qubit, with its environment, can symbolically be written as follows,

$$|Q_0\rangle|e_0\rangle \overset{U}{\longmapsto} \sum |Q_t\rangle|e_t\rangle\,. \tag{5.1}$$

Here, $|Q_0\rangle$ and $|e_0\rangle$ are the initial states of the system and the environment, respectively. At this point, we are assuming that there is no entanglement between them. Since the quantum system is closed, even though it is a combination of two subsystems, according to the laws of quantum mechanics it evolves unitarily. As time passes, however, the mutual interaction produces a final state that is entangled. The state of the combined system at a later time is no longer a product state but an entangled state that we symbolically indicate in (5.1) with the summation symbol.

The whole idea of *quantum error correction* (*QEC*) is precisely to detect and correct the changes occurring in $|Q_0\rangle$ due to the interaction with the environment. Only with such a theory can quantum computer scientists guarantee that large-scale quantum computers will ever be useful.

5.1 Entanglement with the Environment

When a classical bit interacts with its environment, for example, when it is transferred through a noisy channel (actually, all realistic channels are noisy to some extent), the only effect the environment can have on the bit is to flip it. That is, if the bit sent is b, $\bar{b}$ may be received. This is the only type of error that must be taken into account on a classical computing device. The way the environment interacts with a qubit is more complex. Moreover, the environment not only modifies the qubit, but in return it is affected by its interaction with the qubit.

Suppose that an instant before they start interacting, the qubit is in its most general state $|q\rangle = \alpha_0|0\rangle + \alpha_1|1\rangle$ and the environment is in the state $|e\rangle$. At this point, the composite system, single qubit plus environment, is not an entangled state; that is, it is simply described by the tensor product $|q\rangle|e\rangle$. Now, if we denote by U the unitary transformation associated with the evolution of the interacting system, after a certain period of time we will have that

$$U(|0\rangle|e\rangle) = |0\rangle|e_{00}\rangle + |1\rangle|e_{01}\rangle\,, \tag{5.2}$$
$$U(|1\rangle|e\rangle) = |0\rangle|e_{10}\rangle + |1\rangle|e_{11}\rangle\,; \tag{5.3}$$

or, in full form,

$$\begin{aligned} U(|q\rangle|e\rangle) &= U\big((\alpha_0|0\rangle + \alpha_1|1\rangle)|e\rangle\big) = \alpha_0 U(|0\rangle|e\rangle + \alpha_1 U(|1\rangle|e\rangle \\ &= \alpha_0(|0\rangle|e_{00}\rangle + |1\rangle|e_{01}\rangle) + \alpha_1(|0\rangle|e_{10}\rangle + |1\rangle|e_{11}\rangle) \\ &= \sum_{i,j} \alpha_i|j\rangle|e_{ij}\rangle\,. \end{aligned} \tag{5.4}$$

Now, since any 2×2 matrix acting on a single qubit can be written as a linear combination of the Pauli operators I, X, Y, Z, we can rewrite this expression in a

more convenient form,

$$\begin{aligned} U(|q\rangle|e\rangle) &= I|q\rangle|e_I\rangle + X|q\rangle|e_X\rangle + Y|q\rangle|e_Y\rangle + Z|q\rangle|e_Z\rangle \\ &= \sum_A \sigma_A|q\rangle|e_A\rangle\,, \end{aligned} \tag{5.5}$$

where $A = I, X, Y, Z$ and, as usual, we use the shorthand notation $\sigma_A = \sigma_A \otimes I$.

Exercise 5.1. How are the formulas (5.4) and (5.5) related?

Suppose now that we have a 2 qubit $|q_2\rangle$ interacting with its environment $|e\rangle$. The entangled system is described by the state vector

$$\begin{aligned} U(|q_2\rangle|e\rangle) &= U\big((\alpha_{00}|0\,0\rangle + \alpha_{01}|0\,1\rangle + \alpha_{10}|1\,0\rangle + \alpha_{11}|1\,1\rangle)|e\rangle\big) \\ &= I\,I\,|q_2\rangle|e_{II}\rangle + I\,X\,|q_2\rangle|e_{IX}\rangle + I\,Y\,|q_2\rangle|e_{IY}\rangle + I\,Z\,|q_2\rangle|e_{IZ}\rangle \\ &\quad + X\,I\,|q_2\rangle|e_{XI}\rangle + X\,X\,|q_2\rangle|e_{XX}\rangle + X\,Y\,|q_2\rangle|e_{XY}\rangle + X\,Z\,|q_2\rangle|e_{XZ}\rangle \\ &\quad + Y\,I\,|q_2\rangle|e_{YI}\rangle + Y\,X\,|q_2\rangle|e_{YX}\rangle + Y\,Y\,|q_2\rangle|e_{YY}\rangle + Y\,Z\,|q_2\rangle|e_{YZ}\rangle \\ &\quad + Z\,I\,|q_2\rangle|e_{ZI}\rangle + Z\,X\,|q_2\rangle|e_{ZX}\rangle + Z\,Y\,|q_2\rangle|e_{ZY}\rangle + Z\,Z\,|q_2\rangle|e_{ZZ}\rangle\,. \end{aligned}$$

Or, more simply,

$$U(|q_2\rangle|e\rangle) = \sum_{A,B} \sigma_A\,\sigma_B|q_2\rangle|e_{AB}\rangle\,, \tag{5.6}$$

It is clear that for a 3 qubit,

$$U(|q_3\rangle|e\rangle) = \sum_{A,B,C} \sigma_A\,\sigma_B\,\sigma_C|q_3\rangle|e_{ABC}\rangle\,, \tag{5.7}$$

and for an n qubit,

$$U(|Q\rangle|e\rangle) = \sum_{A_1,\ldots,A_n} \sigma_{A_1}\ldots\sigma_{A_n}|Q\rangle|e_{A_1\ldots A_n}\rangle\,. \tag{5.8}$$

Since writing all the subscripts can easily become cumbersome, the following notation is usually used,

$$E_A = \sigma_{A_1}\ldots\sigma_{A_n}\,, \tag{5.9}$$

where $A = 1, \ldots, 4^n$. These new objects are referred as *error operators*. Accordingly, the basis state vectors of the environment's Hilbert space are denoted

$$|e_A\rangle = |e_{A_1\ldots A_n}\rangle\,. \tag{5.10}$$

Employing this new notation,

$$U(|Q\rangle|e\rangle) = \sum_A E_A|Q\rangle|e_A\rangle\,. \tag{5.11}$$

The whole goal of QEC is to identify these E_A's and reverse their action. For instance, the equation (5.5) is telling us that, due to its interaction with the environment, a single qubit can stay unaffected ($\sigma_A = I$) but at the same time it is prone to suffer from a bit flit ($\sigma_A = X$), a phase flip ($\sigma_A = Z$) and a combination of the two ($\sigma_A = Y$; remember that $Y = iXZ$).

Box 5.1. Open quantum systems.

In the previous sections we treated the qubits as almost perfectly isolated quantum systems. We have only allowed them to interact with the gates, which are also perfect quantum objects in the sense that they act on the qubits as unitary transformations. However, the truth is that quantum computers are not perfect quantum systems. For example, the medium through which the qubits propagate to go from one gate to the other can affect the qubit we want to transmit. In order to built real quantum computers, we need to deal with these undesirable situations. The quantum mechanical subdiscipline dealing with this type of phenomena is called "open quantum systems". Since this subject is not usually part of a conventional quantum mechanics course, we will be very brief in our discussion.

The idea, thus, is to provide a mathematical description of the qubit where it no longer behaves as an isolated quantum system with state vector evolving unitarily, but as part of a larger quantum mechanical system that includes other quantum objects affecting it. These external elements are generally called the *environment*. For example, in the transmission of a qubit, the environment may be the medium through which it propagates. The mathematical description of the qubit interacting with its environment is given by the so called *density operator* or *density matrix formalism*. In it, a quantum system is not described by a unit state vector $|\Psi\rangle$, but by a *density operator*, or *density matrix*, defined by $\rho_\Psi = |\Psi\rangle\langle\Psi|$. If the quantum system is a composite system, say a qubit and its environment, this approach allows us to understand the evolution of each of its interacting subsystems, in particular the qubit.

To begin with, let us assume that at some initial time the qubit and the environment are not interacting (to be more precise, that they have never interacted). The composite system $|\Psi_0\rangle$ is thus simply described by the product state $|Q_0\rangle|e_0\rangle$, where $|Q_0\rangle$ and $|e_0\rangle$ are the qubit and the environment initial state vectors, respectively. After some time interacting, the composite state evolves into an entangled state $|\Psi_t\rangle$,

$$|\Psi_0\rangle = |Q_0\, e_0\rangle \mapsto |\Psi_t\rangle = \sum_A E_A|Q_0\, e_A\rangle\,, \tag{5.12}$$

where we have used the shorthand notation $E_A = E_A \otimes I$ and the $|e_A\rangle$'s form a basis for the Hilbert space of all possible final states of the environment. The E_A's are the so called *error operators*. The description of the evolution of the qubit in terms of the density operator formalism is as follows. Since the initial state of the qubit is $|Q_0\rangle$, the density operator is

$$\rho_{Q_0} = |Q_0\rangle\langle Q_0|\,. \tag{5.13}$$

Similarly, the density operator of the final composite system is

$$\begin{aligned}\rho_{\Psi_t} = |\Psi_t\rangle\langle\Psi_t| &= \sum_{A,A'} E_A|Q_0\, e_A\rangle\langle Q_0\, e_{A'}|E_{A'}^\dagger \\ &= \sum_{A,A'} E_A|Q_0\rangle\langle Q_0|E_{A'}^\dagger \otimes |e_A\rangle\langle e_{A'}|\,. \end{aligned}\tag{5.14}$$

The density operator corresponding to the final state of the qubit is somehow contained within ρ_{Ψ_t}. It is called the *reduced density operator* and it is given by

$$\begin{aligned}\rho_{Q_t} &= \mathrm{tr}_e \rho_{\Psi_t} = \mathrm{tr}_e |\Psi_t\rangle\langle\Psi_t| = \mathrm{tr}_e \sum_{A,A'} E_A |Q_0\rangle\langle Q_0| E_{A'}^\dagger \otimes |e_A\rangle\langle e_{A'}| \\ &= \sum_{A,A'} E_A |Q_0\rangle\langle Q_0| E_{A'}^\dagger \mathrm{tr}_e |e_A\rangle\langle e_{A'}| = \sum_{A,A'} E_A |Q_0\rangle\langle Q_0| E_{A'}^\dagger \delta_{AA'} \\ &= \sum_A E_A \rho_{Q_0} E_A^\dagger \,. \end{aligned} \tag{5.15}$$

That is, if the initial state of the qubit is $\rho_{Q_0} = |Q_0\rangle\langle Q_0|$, after some time interacting with the environment, it will evolve into the state $\rho_{Q_t} = \sum_A E_A \rho_{Q_0} E_A^\dagger$.

5.2 Classical Error Correction

Classical computers are also subject to undesirable perturbations arising from their interactions with the environment. Sometimes, for example, we want to send a bit b and it turns out that at the other end of the wire a $\bar{b}$ is received. We review here the *classical repetition code*, one of the multiple ways computer scientists have invented to protect classical information from the destructive effects of the environment. In the last pages, we will see how a similar procedure can be applied to protect quantum information.

Suppose we want to communicate a bit b through a noisy channel and we know that there is a small probability $p \ll 1$ for the bit of getting flipped to $\bar{b}$,

$$P(\bar{b}) = p \,, \qquad P(b) = (1-p) \,. \tag{5.16}$$

Since $p \ll 1$, it follows that $P(b)/P(\bar{b}) \gg 1$.

The repetition code instructs us to send multiple copies of b if we want to decrease the probability of receiving the wrong information. For example, instead of one bit b, one can send three copies of b, that is, we send $b\,b\,b$ rather than b. If every single bit in the string $b\,b\,b$ can get flipped with probability p, we can receive three, two, one or no b's. The corresponding probabilities are as follows,

$$P(3b, 0\bar{b}) = (1-p)^3 \,, \tag{5.17}$$

$$P(2b, 1\bar{b}) = 3(1-p)^2 p \,, \tag{5.18}$$

$$P(1b, 2\bar{b}) = 3(1-p) p^2 \,, \tag{5.19}$$

$$P(0b, 3\bar{b}) = p^3 \,. \tag{5.20}$$

From here we deduce that

$$\frac{P(3b, 0\bar{b})}{P(2b, 1\bar{b})} = \frac{(1-p)^3}{3(1-p)^2 p} = \frac{1}{3}\frac{P(b)}{P(\bar{b})} \,. \tag{5.21}$$

That is, if we send three b's instead of one, the probability of receiving a string with one bit flipped is reduced by a third. Moreover, and this is what is really advantageous about using the repetition code, the relative probability for two bits to get flipped at the same time is

$$\frac{P(3b, 0\bar{b})}{P(1b, 2\bar{b})} = \frac{(1-p)^3}{3(1-p)p^2} = \frac{1}{3}\Big(\frac{P(b)}{P(\bar{b})}\Big)^2 . \tag{5.22}$$

Therefore, the probability for two bits to get flipped simultaneously is very small. It is even smaller for three bits,

$$\frac{P(3b, 0\bar{b})}{P(0b, 3\bar{b})} = \frac{(1-p)^3}{p^3} = \Big(\frac{P(b)}{P(\bar{b})}\Big)^3 . \tag{5.23}$$

Exercise 5.2. To fix the ideas, substitute $p = 10$ and $p = 100$ in the previous example.

Exercise 5.3. Generalize this discussion to a classical repetition code of N bits. Consider both, an odd and an even number of repetitions.

From the previous analysis, we arrive at the following conclusion: if we receive two or three b's, is because the original bit string was $b\,b\,b$. In other words, we apply the *majority rule*. Of course, we could also have received two or three b's when the original message was $\bar{b}\,\bar{b}\,\bar{b}$; however, this is so unlikely that we simply ignore these possibilities.

Exercise 5.4. For a repetition code of N bits, what is the maximum number of flips that can occur for the code to give a correct result?

Hence, we will assume that, if we send the bit string $b\,b\,b$, we can receive $b_1\,b_2\,b_3$, where at most one of the b_i's will be flipped, i.e., $b_1 = \bar{b}$, $b_2 = \bar{b}$ or $b_3 = \bar{b}$. Equivalently, we can say that out of the three initial bits, at least two will remain unchanged: $b_1 = b_2 = b_3 = b$, $b_1 = b_2 = b \neq b_3$, $b_1 = b_3 = b \neq b_2$ and $b_2 = b_3 = b \neq b_1$. The question now is: how do we know if the bits have been corrupted or not? Of course, we can measure them to see if their values are b or $\bar{b}$. But, we can also use the following alternative method that does not require a direct measurement of the bits. It only checks whether two bits have the same or opposite values. This procedure, generally called *parity check*, works as follows:

$$\begin{aligned}
&\text{if } b_1 = b_2\,,\ b_3 = b_1 = b_2 \Rightarrow b_1\,b_2\,b_3 = b\,b\,b\,,\\
&\text{if } b_1 = b_2\,,\ b_3 \neq b_1 = b_2 \Rightarrow b_1\,b_2\,b_3 = b\,b\,\bar{b}\,,\\
&\text{if } b_1 = b_3\,,\ b_2 \neq b_1 = b_3 \Rightarrow b_1\,b_2\,b_3 = b\,\bar{b}\,b\,,\\
&\text{if } b_2 = b_3\,,\ b_1 \neq b_2 = b_3 \Rightarrow b_1\,b_2\,b_3 = \bar{b}\,b\,b\,.
\end{aligned}$$

After detecting which of the bits has been flipped — if any — we reverse it to its original value by applying to it a classical NOT gate.

5.3 Generalities on QEC Codes

Before the first quantum error-correcting codes were invented in the mid-nineties, it was thought that quantum computers were impossible to realize in practice due to the destructive nature of the interaction with the environment. Today, quantum error correcting-codes are known to exist and QEC is a well-established subfield of quantum computing. The general procedure we will follow here is summarized in the following steps:

1. Starting with an n qubit state vector $|Q\rangle$, we create an extended product state by simply adding m *ancillary qubits* (or *ancillas*),

$$|Q\rangle \longmapsto |Q\rangle|0\ldots0\rangle = |Q\rangle_{anc}\,. \tag{5.24}$$

   The ancillary qubits are added because we want to use a quantum repetition code inspired by the classical version discussed in the previous subsection.

2. We then encode the information contained in the original qubit $|Q\rangle$ in the extended state $|Q\rangle_{anc}$. This is done by acting with a unitary transformation U_{enc} on the extended state created in Step 1,

$$|Q\rangle_{anc} \longmapsto U_{enc}|Q\rangle_{anc} = |Q\rangle_{enc}\,. \tag{5.25}$$

   Of course, we need to find out the quantum circuit built from elementary gates that implements the unitary U_{enc}. For the quantum repetition code we will discuss here, this step is rather easy.

3. At this point, the error occurs:

$$|Q\rangle_{enc} \longmapsto E|Q\rangle_{enc} = |Q\rangle_{E}\,. \tag{5.26}$$

   I said "error", and not "errors", because, as for the classical repetition code, we will assume that the probability for two errors to occur at the same time is negligible.

4. Here comes the difficult part. We must design a quantum circuit R that detects and corrects the error,

$$|Q\rangle_{E} \longmapsto R|Q\rangle_{E} = |Q\rangle_{R}\,. \tag{5.27}$$

   In fact, since we cannot measure directly the qubits without destroying the superposition of states, the error is detected indirectly; for example, as we will see, by parity check.

5. We then decode the encoded state by undoing what the encoding operator did,

$$|Q\rangle_{R} \longmapsto U_{dec}|Q\rangle_{R} = |Q\rangle_{anc}\,. \tag{5.28}$$

   Since $U_{dec} = U_{enc}^{-1}$, this is telling us that we need to built another circuit similar to the one corresponding to U_{enc} but performing the inverse operation.

6. Finally, we get rid of the ancillary qubits and recover the original state vector $|Q\rangle$,

$$|Q\rangle_{anc} \longmapsto |Q\rangle\,. \tag{5.29}$$

5.4 Single Qubit Error Correction

Given a single qubit with state vector $|q\rangle$ and two ancillary qubits prepared in the computational basis state $|0\rangle$, we let them pass through the following circuit,

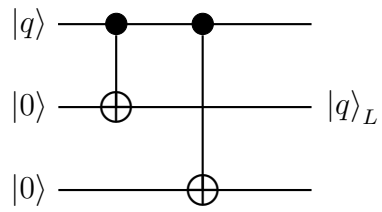

Fig. 41. Production of the logical qubit.

The outgoing state is,

$$|q\rangle_{anc} = |q\rangle|0\rangle|0\rangle = \alpha_0|0\,0\,0\rangle + \alpha_1|1\,0\,0\rangle \longmapsto \alpha_0|0\,0\,0\rangle + \alpha_1|1\,1\,1\rangle \equiv |q\rangle_L\,. \tag{5.30}$$

The state $|q\rangle_L$, corresponding to $|Q\rangle_{enc}$ in (5.25), is called the *logical qubit* to distinguish it from the *physical qubit* $|q\rangle$ we had originally.

Note that, when we added the two ancillas $|0\rangle|0\rangle$ to the qubit $|q\rangle$, we extended the Hilbert space from two to eight dimensions,

$$|q\rangle \in \mathcal{H}_q \cong \mathbb{C}^2\,, \qquad |q\rangle_{anc} \in \mathcal{H}_{q_{anc}} \cong \mathbb{C}^8\,. \tag{5.31}$$

Since it is in this extended Hilbert space that most of the error-correcting code we will discuss operates, it is worth mentioning some of its most relevant properties.

Two basis vectors of the Hilbert space $\mathcal{H}_{q_{anc}}$ are $|0\,0\,0\rangle$ and $|1\,1\,1\rangle$. The other six basis vectors can be chosen to be, as usual, $|0\,0\,1\rangle$, $|0\,1\,0\rangle$, $|0\,1\,1\rangle$, $|1\,0\,0\rangle$, $|1\,0\,1\rangle$ and $|1\,1\,0\rangle$. Any vector $|\chi\rangle \in \mathcal{H}_{q_{anc}}$ will then be a linear combination of these basis vectors,

$$\begin{aligned}|\chi\rangle &= \alpha_{000}|0\,0\,0\rangle + \alpha_{001}|0\,0\,1\rangle + \alpha_{010}|0\,1\,0\rangle + \alpha_{011}|0\,1\,1\rangle \\ &\quad + \alpha_{100}|1\,0\,0\rangle + \alpha_{101}|1\,0\,1\rangle + \alpha_{110}|1\,1\,0\rangle + \alpha_{111}|1\,1\,1\rangle \\ &= \big(\alpha_{000}|0\,0\,0\rangle + \alpha_{111}|1\,1\,1\rangle\big) + \big(\alpha_{100}|1\,0\,0\rangle + \alpha_{011}|0\,1\,1\rangle\big) \\ &\quad + \big(\alpha_{010}|0\,1\,0\rangle + \alpha_{101}|1\,0\,1\rangle\big) + \big(\alpha_{001}|0\,0\,1\rangle + \alpha_{110}|1\,1\,0\rangle\big)\,.\end{aligned} \tag{5.32}$$

The vectors in parentheses are contained in four mutually orthogonal subspaces of $\mathcal{H}_{q_{anc}}$,

$$\begin{aligned}\mathcal{F}_0 &= \{|0\,0\,0\rangle, |1\,1\,1\rangle\}\,, \qquad \mathcal{F}_1 = \{|1\,0\,0\rangle, |0\,1\,1\rangle\}\,, \\ \mathcal{F}_2 &= \{|0\,1\,0\rangle, |1\,0\,1\rangle\}\,, \qquad \mathcal{F}_3 = \{|0\,0\,1\rangle, |1\,1\,0\rangle\}\,.\end{aligned} \tag{5.33}$$

Note that we have chosen the basis vectors of the subspace $\mathcal{F}_1$ so that they correspond to the basis vectors of $\mathcal{F}_0$ with the first bit flipped, that is,

$$|1\,0\,0\rangle = X\,I\,I\,|0\,0\,0\rangle\,, \qquad |0\,1\,1\rangle = X\,I\,I\,|1\,1\,1\rangle\,. \tag{5.34}$$

Similar, of course, for the basis vectors of $\mathcal{F}_2$ and $\mathcal{F}_3$.

It follows, then, that any vector in $\mathcal{H}_{q_{anc}}$ can be written as

$$\begin{aligned}|\chi\rangle = I\,I\,I\left(\alpha_{000}|0\,0\,0\rangle + \alpha_{111}|1\,1\,1\rangle\right) + X\,I\,I\left(\alpha_{100}|0\,0\,0\rangle + \alpha_{011}|1\,1\,1\rangle\right)\\ + I\,X\,I\left(\alpha_{010}|0\,0\,0\rangle + \alpha_{101}|1\,1\,1\rangle\right) + I\,I\,X\left(\alpha_{001}|0\,0\,0\rangle + \alpha_{110}|1\,1\,1\rangle\right). \quad (5.35)\end{aligned}$$

Now that we understand the basic geometry of the Hilbert space $\mathcal{H}_{q_{anc}}$, let us consider the effect of the environment. For a single qubit, we saw in (5.5) that,

$$|q\rangle \longmapsto \sum_A \sigma_A|q\rangle\,. \quad (5.36)$$

However, since we have encoded the information of the single qubit $|q\rangle$ in the logical qubit $|q\rangle_L$ given in (5.30), we have now to evaluate the effect of the environment on each physical qubit of $|q\rangle_L$.
In general, several errors can simultaneously occur on each physical qubit,

$$|q\rangle_L = \sum_i \alpha_i|i\rangle|i\rangle|i\rangle \longmapsto \sum_i \alpha_i \sum_A \sigma_A|i\rangle \sum_B \sigma_B|i\rangle \sum_C \sigma_C|i\rangle\,, \quad (5.37)$$

where $\sigma_A, \sigma_B, \sigma_C = I, X, Y, Z$. But, since we want to consider at most one error per physical qubit,

$$|q\rangle_L \longmapsto \sum_i \alpha_i|i\,i\,i\rangle + \sum_i \alpha_i\sigma_a|i\rangle|i\rangle|i\rangle + \sum_i \alpha_i|i\rangle\sigma_b|i\rangle|i\rangle + \sum_i \alpha_i|i\rangle|i\rangle\sigma_c|i\rangle\,.$$

where $\sigma_a, \sigma_b, \sigma_c = X, Y, Z$ and the first summation symbol takes into account the possibility that nothing happens to the qubits. This expression is still too general. In fact, it allows for errors of different nature and we are only interested in errors of the same type. Hence,

$$|q\rangle_L \longmapsto \sum_i \alpha_i|i\,i\,i\rangle + \sum_i \alpha_i\sigma_a|i\rangle|i\rangle|i\rangle + \sum_i \alpha_i|i\rangle\sigma_a|i\rangle|i\rangle + \sum_i \alpha_i|i\rangle|i\rangle\sigma_a|i\rangle\,.$$

Finally, if we consider bit-flip errors, that is, $\sigma_a = X$, the corrupted qubit will be described by the following state vector

$$\begin{aligned}&|q\rangle_L \longmapsto |q\rangle_E\\ &= \sum_i \alpha_i|i\,i\,i\rangle + \sum_i \alpha_i X|i\rangle|i\rangle|i\rangle + \sum_i \alpha_i|i\rangle X|i\rangle|i\rangle + \sum_i \alpha_i|i\rangle|i\rangle X|i\rangle\,. \quad (5.38)\end{aligned}$$

More explicitly,

$$\begin{aligned}|q\rangle_E = \alpha_0|0\,0\,0\rangle + \alpha_1|1\,1\,1\rangle + \alpha_0|1\,0\,0\rangle + \alpha_1|0\,1\,1\rangle\\ + \alpha_0|0\,1\,0\rangle + \alpha_1|1\,0\,1\rangle + \alpha_0|0\,0\,1\rangle + \alpha_1|1\,1\,0\rangle\,. \quad (5.39)\end{aligned}$$

Remember that, actually, we do not know which physical qubit of the logical qubit has been flipped. The goal is to identify and correct it. The circuit that does this is the following:

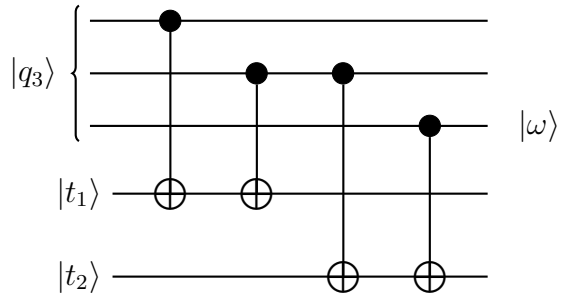

Fig. 42. Three-qubit parity check.

As a matter of fact, the auxiliary qubits $|t_1\rangle$ and $|t_2\rangle$ introduced in Figure 42 need to be in the state $|0\rangle$. However, for practice, let us first consider the most general case,

$$
\begin{aligned}
|q_3\rangle|t_1\rangle|t_2\rangle &= \sum_{i,j,k,l',m'} \alpha_{ijk} t_{1,l'} t_{2,m'} |i\,j\,k\rangle|l'\rangle|m'\rangle \\
&\xmapsto{\text{CNOT}_{11}} \sum_{i,j,k,l',m'} \alpha_{ijk} t_{1,l'} t_{2,m'} |i\,j\,k\rangle|l' \oplus i\rangle|m'\rangle \\
&\xmapsto{\text{CNOT}_{21}} \sum_{i,j,k,l',m'} \alpha_{ijk} t_{1,l'} t_{2,m'} |i\,j\,k\rangle|l' \oplus i \oplus j\rangle|m'\rangle \\
&\xmapsto{\text{CNOT}_{22}} \sum_{i,j,k,l',m'} \alpha_{ijk} t_{1,l'} t_{2,m'} |i\,j\,k\rangle|l' \oplus i \oplus j\rangle|m' \oplus j\rangle \\
&\xmapsto{\text{CNOT}_{32}} \sum_{i,j,k,l',m'} \alpha_{ijk} t_{1,l'} t_{2,m'} |i\,j\,k\rangle|l' \oplus i \oplus j\rangle|m' \oplus j \oplus k\rangle\,.
\end{aligned}
$$

Now, since we want $|t_1\rangle = |t_2\rangle = |0\rangle$, we substitute $t_{1,0} = t_{2,0} = 0$ and $t_{1,1} = t_{2,1} = 1$ in the previous result, giving

$$
\begin{aligned}
|q_3\rangle|0\rangle|0\rangle &\mapsto \sum_{i,j,k} \alpha_{ijk} |i\,j\,k\rangle|1 \oplus i \oplus j\rangle|1 \oplus j \oplus k\rangle \\
&= \alpha_{000}|0\,0\,0\rangle|1 \oplus 0 \oplus 0\rangle|1 \oplus 0 \oplus 0\rangle + \alpha_{001}|0\,0\,1\rangle|1 \oplus 0 \oplus 0\rangle|1 \oplus 0 \oplus 1\rangle \\
&\quad + \alpha_{010}|0\,1\,0\rangle|1 \oplus 0 \oplus 1\rangle|1 \oplus 1 \oplus 0\rangle + \alpha_{011}|0\,1\,1\rangle|1 \oplus 0 \oplus 1\rangle|1 \oplus 1 \oplus 0\rangle \\
&\quad + \alpha_{100}|1\,0\,0\rangle|1 \oplus 1 \oplus 0\rangle|1 \oplus 0 \oplus 0\rangle + \alpha_{101}|1\,0\,1\rangle|1 \oplus 1 \oplus 0\rangle|1 \oplus 0 \oplus 1\rangle \\
&\quad + \alpha_{110}|1\,1\,0\rangle|1 \oplus 1 \oplus 0\rangle|1 \oplus 1 \oplus 0\rangle + \alpha_{111}|1\,1\,1\rangle|1 \oplus 1 \oplus 1\rangle|1 \oplus 1 \oplus 1\rangle \\
&= \big(\alpha_{010}|0\,1\,0\rangle + \alpha_{101}|1\,0\,1\rangle\big)|0\,0\rangle + \big(\alpha_{100}|1\,0\,0\rangle + \alpha_{011}|0\,1\,1\rangle\big)|0\,1\rangle \\
&\quad + \big(\alpha_{001}|0\,0\,1\rangle + \alpha_{110}|1\,1\,0\rangle\big)|1\,0\rangle + \big(\alpha_{000}|0\,0\,0\rangle + \alpha_{111}|1\,1\,1\rangle\big)|1\,1\rangle \\
&= I\,X\,I\big(\alpha_{010}|0\,0\,0\rangle + \alpha_{101}|1\,1\,1\rangle\big)|0\,0\rangle + X\,I\,I\big(\alpha_{100}|0\,0\,0\rangle + \alpha_{011}|1\,1\,1\rangle\big)|0\,1\rangle \\
&\quad + I\,I\,X\big(\alpha_{001}|0\,0\,0\rangle + \alpha_{110}|1\,1\,1\rangle\big)|1\,0\rangle + I\,I\,I\big(\alpha_{000}|0\,0\,0\rangle + \alpha_{111}|1\,1\,1\rangle\big)|1\,1\rangle\,.
\end{aligned}
$$

Finally, since the arbitrary qubit $|q_3\rangle$ used above is indeed $|q\rangle_E$ given explicitly in (5.38), we must take

$$
\begin{aligned}
\alpha_{010} = \alpha_{100} = \alpha_{001} = \alpha_{000} &= \alpha_0\,, \\
\alpha_{101} = \alpha_{011} = \alpha_{110} = \alpha_{111} &= \alpha_1\,.
\end{aligned}
$$

After substituting, we get

$$|q\rangle_E|0\,0\rangle \longmapsto |q\rangle_R \tag{5.40}$$
$$= I\,X\,I\,|q\rangle_L|0\,0\rangle + X\,I\,I\,|q\rangle_L|0\,1\rangle + I\,I\,X\,|q\rangle_L|1\,0\rangle + I\,I\,I\,|q\rangle_L|1\,1\rangle\,.$$

We now measure the auxiliary qubits and do the following:

$$\begin{array}{l}\text{if the measurement gives } |0\rangle|0\rangle, \text{ we apply } I\,X\,I\,,\\ \text{if the measurement gives } |0\rangle|1\rangle, \text{ we apply } X\,I\,I\,,\\ \text{if the measurement gives } |1\rangle|0\rangle, \text{ we apply } I\,I\,X\,,\\ \text{if the measurement gives } |1\rangle|1\rangle, \text{ we apply } I\,I\,I\,.\end{array} \tag{5.41}$$

Regardless of the measurement outcome, the procedure (5.41) will always result in the state vector $|q\rangle_L$. We finally get rid of the ancillary qubits by using the following circuit,

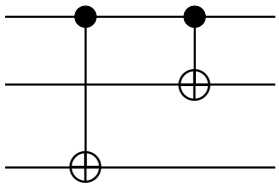

Fig. 43. Decoding gate.

Indeed, the outgoing state is,

$$|q\rangle_L = \alpha_0|0\,0\,0\rangle + \alpha_1|1\,1\,1\rangle \longmapsto \alpha_0|0\rangle + \alpha_1|1\rangle = |q\rangle\,. \tag{5.42}$$

We have provided a complete description of the bit flip error-correcting code. However, as equation (5.37) shows, many other errors can occur to the logical qubit $|q\rangle_L$. The treatment of the general case will be the subject of future notes.

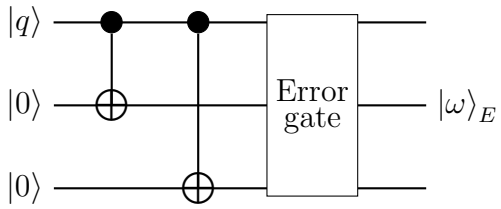

Fig. 44. The error gate.

6 Bibliography

If you think you need additional supporting material, maybe because an idea or calculation in my notes is not clear enough, consult the book by Kaye *et al.* It is a bit more elementary and it is very well-written. A textbook at the same level as these notes is Nielsen & Chuang, which is the classic reference on the subject. In addition to these books, you may find useful the free online resources I list below. In particular, the notes by Preskill are worth studying, especially because they were

written from the viewpoint of a theoretical physicist and the video lectures can be found online. Finally, I highly recommend that you watch the online lectures by Nathan Wiebe.

[1] S. Aaronson, "Introduction to Quantum Information Science: Lecture Notes".

[2] A. Ekert, "Introduction to Quantum Computation".

[3] S. Girvin, "Introduction to Quantum Error Correction and Fault Tolerance".

[4] R. Jozsa, "Quantum Information and Computation".

[5] P. Kaye, R. Laflamme & M. Mosca, *An Introduction to Quantum Computing.*

[6] E. Knill et al., "Introduction to Quantum Information Processing".

[7] S. Lloyd, "Quantum Information Science".

[8] M. Mosca, "Quantum Algorithms".

[9] M. Nielsen & I. Chuang, *Quantum Computation and Quantum Information.*

[10] J. Preskill, "Quantum Computation: Lecture Notes".

[11] A. Steane, "Quantum Computing".

[12] R. de Wolf, "Quantum Computing: Lecture Notes".

Index

R_l gate, 22
$\pi/8$ gate, 15
$\sqrt{\text{NOT}}$ gate, 15
k-local Hamiltonians, 53

Algorithm, 33
Ancillary qubits, 59

Balanced function, 35, 38
Bell basis, 47
Bell states, 47
Binary system, 4
Bit, 4
Bit flip gate, 13
Bit string, 4
Black box, 35
Bloch sphere, 6
Boolean function, 10

CCNOT gate, 28
Circuit diagram (classical), 10
Circuit model of computation (classical), 10
Classical repetition code, 57
Clifford gate, 15
CNOT gate, 22
Computational basis, 5, 7, 9
Computer, 3
Constant function, 34, 38
Control qubit, 20
Control qubit, first, 28
Control qubit, second, 28
Controlled gate, 20
Controlled-U gate, 20
Controlled-V gate, 20
Controlled-X gate, 22
Controlled-Z gate, 22
Controlled-controlled NOT gate, 28
Controlled-NOT gate, 22
Controlled-SWAP gate, 24
Cryptography, 41
CSWAP gate, 24

Density operator (matrix), 56
Deutsch's algorithm, 34
Deutsch-Jozsa's algorithm, 38

Entangled state, 8
Environment, 56
EPR pair, 47
Error operator, 55, 56

Fourier basis, 45

Gate (classical logic), 10
Gate (quantum), 11, 18
GHZ state, 8

Hadamard basis, 5
Hadamard gate, 14, 16
Hamiltonian operator, 49
Hamiltonian simulation, 50

Logical qubit, 60

Majority rule, 58
Measuring qubit, 31
Multiple or n qubit, 9

NOT gate (classical), 10
NOT gate (quantum), 13

Oracle, 35

Parity check, 58
Pauli group, 52
Pauli matrices, 12
Pauli operators, 52
Period, 43
Phase flip gate, 15
Physical qubit, 60
Product formula, 53
Product formula simulation, 53
Product state, 8

Quantum error correction, QEC, 54
Quantum Fourier transform, QFT, 45
Quantum phase estimation, 31
Quantum simulation, 50
Quantum teleportation, 48
Qubit, 4
Query complexity, 35

Reduced density operator, 57
Relative phase gate, 15

S gate, 15
Schrödinger's equation, 49
Shor's algorithm, 41

Single qubit, 1 qubit, 4
Single-qubit gate, 11
Superdense coding, 48
SWAP gate, 23

T gate, 15
Target qubit, 20
Tensor product, 7, 19
Time evolution operator, 49
Toffoli gate, 28

Unitary, 11
Universal set of gates (classical), 10

XOR oracle, 35

Notes for a Second Course on Quantum Computing for Physicists

Oswaldo Zapata

Abstract

In these notes I continue the course on quantum computing I started in arXiv:2306.09388. I begin by introducing the density operator formalism as an alternative to the state vector formalism and focus on entangled quantum systems. After a short presentation of information theory, both the classical and quantum versions, I present a couple of quantum algorithms, in particular, the variational quantum eigensolver algorithm, an algorithm that is expected to be implemented in near-term quantum computers. I conclude with some advanced topics on quantum error correction that I did not cover in my precedent monograph. These notes are intended for students and professional physicists who have already studied my earlier notes or a similar introduction to quantum computing.

1 Introduction **2**

2 The Density Operator Formalism **2**
2.1 Density Operators . . . 3
2.2 Multipartite Systems . . . 9
2.3 State Evolution . . . 13
2.4 Measurement . . . 16

3 Information **18**
3.1 Classical Information Theory . . . 20
3.2 Quantum Information Theory . . . 28

4 Algorithms and Secure Communication **33**
4.1 Quantum Phase Estimation . . . 34
4.2 The Variational Quantum Eigensolver . . . 43
4.3 Quantum Cryptography . . . 46
4.3.1 BB84 Protocol . . . 46

5 Error Correction and Fault Tolerance **48**
5.1 Single-Qubit Quantum Channels . . . 48
5.2 Stabilizers Circuits . . . 51
5.3 Stabilizer QEC Codes . . . 56
5.4 Fault-Tolerant QEC . . . 59

6 Bibliography **59**

1 Introduction

These notes are a natural continuation of my previous review article on quantum computing (arXiv:2306.09388, from now on simply referred as QC1). If you studied it, or are familiar with a similar introduction to quantum computing, you should be able to read and understand the present notes with minimum effort. They are not long and I have tried my best to write them in the most pedagogical manner. The many exercises interspersed within the text will help you fix the new concepts and develop your mathematical skills.

In Section 2, I present the density operator formalism and explain how it helps describe subsystems of larger quantum systems. Since information theory, as understood by computer scientists, is not a subject usually contained in a standard physics curriculum, in Section 3 I give a brief overview of classical as well as quantum information theory. Scientists believe that quantum computers will help us understand better many complex systems, in particular atoms, molecules and solids. In Section 4, I present two algorithms that have been developed specifically to deal with these practical situations: the quantum phase estimation algorithm and the variational quantum eigensolver. In this section I also introduce the first ever invented secure communication protocol that uses the principles of quantum mechanics, known as the BB84 protocol. In Section 5, I conclude with a brief discussion on quantum error correction and how the stabilizer theory of quantum mechanics applies to it.

Here is a brief historical summary of the topics discussed in these notes. The density operator formalism was created by John von Neumann in 1929. It is the result of von Neumann's desire to explain realistic situations where the quantum state of a system cannot be known with precision due to its interaction with other quantum systems. Quantum information theory is an adaptation to the quantum world of the seminal ideas introduced in 1948 by Claude Shannon in a classical context. The quantum phase estimation algorithm was proposed by Alexei Kitaev in 1996 and the variational quantum eigensolver algorithm by Alberto Peruzzo and collaborators in 2014. In the mid 90s the concept of fault-tolerant quantum computation was independently formulated by Peter Shor and Alexei Kitaev. Daniel Gottesman, also by the end of the last century, was the first to apply the stabilizer formalism to quantum error correction.

If you want to contact me regarding this paper, maybe you found a couple of typos or simply because you want to share your feedback, please send me an email at zapata.oswaldo@gmail.com. Your comments will be welcomed.

2 The Density Operator Formalism

The mathematical framework we introduce in this section is an alternative to the conventional state vector formalism of quantum mechanics most of us learned in college. The so called *density operator* (or *density matrix*) *formalism* of quantum mechanics was especially developed by John von Neumann to study quantum systems that cannot be considered isolated from the rest of the world. In QC1, we did not need it because most of our systems were isolated or, in the case of interaction with their environments, we did managed to provide a state vector description (as in Chapter 4 on quantum error correction). However, the circuit components created by physicists, such as qubits, gates and measurement apparatuses, are far from ideal

and they always interact with their environments. Accordingly, we now revisit the basic principles of quantum mechanics in this more realistic formalism.

2.1 Density Operators

We start our exposition with two common situations where the description provided by the state vector formalism is incomplete.

To begin with, consider the Bell state

$$|\beta_0\rangle = \frac{1}{\sqrt{2}}\big(|0\rangle|0\rangle + |1\rangle|1\rangle\big) = \frac{1}{\sqrt{2}}\sum_i |i\rangle|i\rangle\,, \tag{2.1}$$

(see QC1, Subsection 4.3). Since the state vector $|\beta_0\rangle$ describes a composite system made of two single qubits, its Hilbert space $\mathcal{H}$ is the tensor product of the individual Hilbert spaces, $|\beta_0\rangle \in \mathcal{H} = \mathcal{H}_q \otimes \mathcal{H}_{q'}$. Of course, we know that in addition to entangled states, such as $|\beta_0\rangle$ itself, the Hilbert space $\mathcal{H}$ contains vectors associated to non-entangled two-qubit systems. That is, if $|q\rangle \in \mathcal{H}_q$ and $|q'\rangle \in \mathcal{H}_{q'}$, there are two-qubit systems for which the state vector is a simple product state, $|q\rangle \otimes |q'\rangle \in \mathcal{H}$. Thus, the Hilbert space $\mathcal{H}$ contains entangled and non-entangled states, of which $|\beta_0\rangle$ is an example of the latter.

The problem with this mathematical description is that, despite the fact that the vector $|\beta_0\rangle$ provides a complete description of the entangled two-qubit system, there is no state vector description of each individual single qubit. You may argue that, we do not need a state vector description of each individual single qubit because, as soon as we observe one qubit in $|i\rangle$, we know with certainty that the other qubit is also in $|i\rangle$. From the empirical point of view, thus, the measurement process renders unnecessary an independent state vector description of each qubit. The previous interpretation is how the entanglement of two single qubits is understood in the context of the state vector formalism. However, from the theoretical point of view, it is clear that this description is incomplete. This shortcoming becomes more apparent when we deal with more complex situations. For example, as we will see next, if we want to understand how a single qubit interacts with its environment.

Suppose a quantum system made of a single qubit surrounded by a large complex system (the environment). The total system, qubit plus environment, is described by a state vector

$$|\Psi\rangle = \sum_{i,j} \alpha_{ij}|i\rangle|e_i\rangle\,. \tag{2.2}$$

Again, as for the Bell pair, we cannot find a state vector description of each subsystem separately. However, similarly, if we measure the qubit in $|i\rangle$, we know that the environment is in the state $|e_i\rangle$. Conversely, if we could measure the state of the environment, the result $|e_i\rangle$ would imply that the qubit is in $|i\rangle$. Unfortunately, by definition, the environment is so complex that we cannot make sense experimentally of the state $|e_i\rangle$. Thus, the operational approach taken above for two entangled single qubits cannot be used here. Therefore, the state vector formalism is rather useless if we want to study two or more complex quantum systems in interaction.

The desire to understand the evolution in time of each individual qubit makes it even more apparent the need for an alternative approach to the state vector formalism. In the case of the Bell state $|\beta_0\rangle$, the interaction of the individual qubits can be modeled by a unitary matrix U represented in the following diagram,

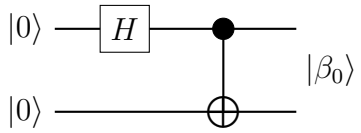

Fig. 1. Creation of the $|\beta_0\rangle$ state.

that is,

$$|0\rangle|0\rangle \xmapsto{H\otimes 1} \xmapsto{\text{CNOT}} U|0\rangle|0\rangle = |\beta_0\rangle\,. \tag{2.3}$$

Something similar happens with the evolution of a single qubit interacting with its environment (see QC1, Subsection 5.2),

$$|q\rangle|e\rangle \xmapsto{U} \sum_{i,j} \alpha_i|j\rangle|e_{ij}\rangle\,. \tag{2.4}$$

However, there is no description of the evolution of each subsystem separately. In fact, it turns out that a state vector description of the time evolution of each subsystem would imply a non-unitary matrix acting on them.

In the following pages we will see how the density operator formalism allows us to describe a subsystem of an entangled quantum system ignoring everything that does not form part of it. Before telling you how we can do that, though, we must present the necessary physical ideas and mathematical tools of the density operator formalism.

Imagine that you are in front of a black box containing a quantum system. From the black box you receive an n qubit and you measure it. The information carried by the qubit is all you know about the system in the box. In the state vector formalism, the situation is described as follows. Before the measurement, the n qubit can be represented by a state vector $|Q\rangle$ in a Hilbert space $\mathcal{H}_Q$ of dimension 2^n. If $\{|Q_{s'}\rangle\}$ is a basis for $\mathcal{H}_Q$, the state vector $|Q\rangle$ can be written as a linear superposition of these basis vectors,

$$|Q\rangle = \sum_{s'=1}^{2^n} \alpha_{s'}|Q_{s'}\rangle\,. \tag{2.5}$$

Now, suppose there is an apparatus that measures 2^n independent observational states of this particular quantum system. Let us denote them by $\{|O_s\rangle\}$ and by $\mathcal{H}_O$ the Hilbert space they span. Quantum mechanics postulates that the probability of measuring the system in the observational state $|O_s\rangle$ is given by

$$P(|O_s\rangle) = p_s = |\langle O_s|Q\rangle|^2 = \Big|\sum_{s'} \alpha_{s'}\langle O_s|Q_{s'}\rangle\Big|^2\,. \tag{2.6}$$

When the two bases $\{|Q_{s'}\rangle\}$ and $\{|O_s\rangle\}$ coincide, the qubit state can be written

$$|Q\rangle = \sum_{s=1}^{2^n} \alpha_s|O_s\rangle\,. \tag{2.7}$$

If, moreover, the basis is orthonormal, that is, $\langle O_s|O_{s'}\rangle = \delta_{ss'}$, for any pair of indices $s, s' = 1, \ldots, 2^n$, the probability of measuring the qubit $|Q\rangle$ in the state $|O_s\rangle$ is

$$p_s = \Big|\sum_{s'} \alpha_{s'}\langle O_s|O_{s'}\rangle\Big|^2 = \Big|\sum_{s'} \alpha_{s'}\delta_{ss'}\Big|^2 = |\alpha_s|^2\,. \tag{2.8}$$

Thus, from an experimental point of view, a quantum system is described by a state vector $|Q\rangle$ in the Hilbert space $\mathcal{H}_O$ of observational states $\{|O_s\rangle\}$. In particular, in the computational basis $\{|x\rangle\}$,

$$|Q\rangle = \sum_x \alpha_x |x\rangle \,, \tag{2.9}$$

and $P\big(|x\rangle\big) = p_x = |\alpha_x|^2$ (see equation (2.27) of QC1).

Note that, this description assumes that we have a complete knowledge of the state of the qubit $|Q\rangle$, that is, that we know all the coefficients α_s. However, in reality, we do not know them; the only thing we can measure are the probabilities p_s. In order to take into consideration this experimental fact and, at the same time, preserve the quantum superposition principle, it is, of course, incorrect to simply substitute α_s by $\sqrt{p_s}$ in (2.7).

The density operator formalism approaches the description of a quantum system from another angle.

Instead of a single quantum system in a black box, now, imagine that you are in front of an infinite number of black boxes, all of which contain identical copies of the same quantum system. Moreover, suppose that every quantum system emits an n qubit. A measurement will give again an observational state $|O_s\rangle$ belonging to the Hilbert space $\mathcal{H}_O$ spanned by all possible outcomes $\{|O_s\rangle\}$. The probability that a qubit was emitted in the state $|O_s\rangle$ is again p_s. The infinite black boxes and the experimental probabilities p_s provide a theoretical model for the quantum system inside the box. The system is, thus, completely characterized by the only experimental data allowed to us: the probability distribution $\{|O_s\rangle, p_s\}$. Note that, there is no probability amplitude α_s, only p_s. In statistical mechanics, an infinite number of macroscopic systems used to model a probabilistic microscopic system is known as a (*statistical*) *ensemble*.

When a statistical ensemble is prepared in such a way that there is only one possible outcome, say $|Q_s\rangle$ with $p_s = 1$, we say that the system is in a *pure state*. On the contrary, when there are at least two possible outcomes, $|Q_{s_1}\rangle$ and $|Q_{s_2}\rangle$ with $p_{s_1} \neq p_{s_2}$, we say that the system is *mixed* (or that it is a mixture of pure states).

In the *density operator formalism* of quantum mechanics, the mathematical description of a quantum system is provided by the so called *statistical* or *density operator*,

$$\rho = \sum_s p_s \, \Pi_s \,, \tag{2.10}$$

where $\Pi_s \colon \mathcal{H}_Q \to \mathcal{H}_Q$ is the projection operator over the observational state $|O_s\rangle$. A convenient notation for the projector Π_s is

$$\Pi_s = |O_s\rangle\langle O_s| \,. \tag{2.11}$$

Using this expression for the projectors, the density operator becomes

$$\rho = \sum_s p_s |O_s\rangle\langle O_s| \,. \tag{2.12}$$

This notation is very practical because now we can easily see how the density operator acts on quantum states in $\mathcal{H}_Q$. In fact, since the density operator is a map $\rho_Q \colon \mathcal{H}_Q \to \mathcal{H}_Q$, we can use the standard vector representation of a quantum system

to see how ρ operates on it. In general, for a state vector $|Q\rangle \in \mathcal{H}_Q$ written as a linear superposition of the basis vectors $\{|Q_{s'}\rangle\}$ of $\mathcal{H}_Q$, we have

$$\begin{aligned}\rho|Q\rangle = \rho \sum_{s'} \alpha_{s'}|Q_{s'}\rangle &= \sum_{s} p_s|O_s\rangle\langle O_s| \sum_{s'} \alpha_{s'}|Q_{s'}\rangle \\ &= \sum_{s,s'} p_s \alpha_{s'}|O_s\rangle\langle O_s|Q_{s'}\rangle \,. \end{aligned} \tag{2.13}$$

Now, if we choose for both, the observational Hilbert space $\mathcal{H}_O$ and the qubit Hilbert space $\mathcal{H}_Q$, the same orthonormal basis $\{|O_s\rangle\}$, the above relation reduces to

$$\rho|Q\rangle = \rho \sum_{s'} \alpha_{s'}|O_{s'}\rangle = \sum_{s,s'} p_s \alpha_{s'}|O_s\rangle\, \delta_{ss'} = \sum_{s} p_s \alpha_s |O_s\rangle \,. \tag{2.14}$$

When the qubit is known to be in an observational state of the apparatus, $|Q\rangle = |O_{s'}\rangle$,

$$\rho|O_{s'}\rangle = \sum_{s} p_s|O_s\rangle\langle O_s|O_{s'}\rangle = \sum_{s} p_s|O_s\rangle\, \delta_{ss'} = p_{s'}|O_{s'}\rangle \,, \tag{2.15}$$

and

$$\langle O_{s'}|\rho|O_{s'}\rangle = p_{s'} \,. \tag{2.16}$$

Exercise 2.1. Show that pure states are the only ones for which $\rho^2 = \rho$.

Having defined the density operator, we now need to explain how the remaining postulates of quantum mechanics, such as time evolution and measurement, must be adapted to this new framework. We will do this in the following pages. Before that, though, let us see some properties of the density operator and consider some simple examples.

The trace of a density operator, that is, the sum of all its diagonal elements, is

$$\operatorname{Tr} \rho = \sum_{s} \langle O_s|\rho|O_s\rangle = \sum_{s} p_s = 1 \,. \tag{2.17}$$

Therefore, the density operator has unit trace. Note that this property corresponds to the normalization condition of the state vector formalism.

That the density operator is Hermitian is also easy to show. Just remember that, by definition, projectors are Hermitian operators, hence,

$$\rho^\dagger = \sum_{s} p_s^* \big(|O_s\rangle\langle O_s|\big)^\dagger = \sum_{s} p_s|O_s\rangle\langle O_s| = \rho \,. \tag{2.18}$$

Moreover, since Hermitian operators are diagonalizable, we can always find a basis of orthonormal eigenvectors $\{e_s\}$ to write a density operator as

$$\rho = \sum_{s} p_s|e_s\rangle\langle e_s| \,. \tag{2.19}$$

This is called the *eigenvalue decomposition* of the density operator because,

$$\rho|e_s\rangle = \sum_{s'} p_{s'}|e_{s'}\rangle\langle e_{s'}|e_s\rangle = \sum_{s'} p_{s'}|e_{s'}\rangle\, \delta_{ss'} = p_s|e_s\rangle \,. \tag{2.20}$$

From here, it follows that

$$p_s = \langle e_s|\rho|e_s\rangle \,. \tag{2.21}$$

Exercise 2.2. Show that

$$p_s = \mathrm{Tr}\left[\rho|e_s\rangle\langle e_s|\right]. \tag{2.22}$$

Finally, because probabilities are equal or greater than zero, it can be proved that density operators are also *positive (semi-definite)*,

$$\langle Q|\rho|Q\rangle \geq 0\,, \tag{2.23}$$

where $|Q\rangle$ is any state vector in $\mathcal{H}_Q$. For instance, if we write $|Q\rangle$ in the orthonormal basis $\{e_s\}$ of $\mathcal{H}_O$,

$$\begin{aligned}\langle Q|\rho|Q\rangle &= \sum_{s,s'} \alpha_s^*\alpha_{s'}\langle e_s|\rho|e_{s'}\rangle = \sum_{s,s'} \alpha_s^*\alpha_{s'}\langle e_s|p_{s'}|e_{s'}\rangle \\ &= \sum_{s,s'} \alpha_s^*\alpha_{s'}p_{s'}\langle e_s|e_{s'}\rangle = \sum_{s,s'} \alpha_s^*\alpha_{s'}p_{s'}\delta_{ss'} \\ &= \sum_{s} \alpha_s^*\alpha_s p_s = \sum_{s} p_s^2 \geq 0\,.\end{aligned} \tag{2.24}$$

Exercise 2.3. Prove that density operators are positive, regardless of the basis chosen for $\mathcal{H}_Q$

To sum up, the density operator of a quantum state has unit trace, is Hermitian and is positive semi-definite. It can be proved that the converse is also true: any unit trace operator, which is Hermitian and positive semi-definite, is the density operator of some quantum state. Since the density operator contains all the physical information we can have about a quantum system, we often say that ρ is the quantum state. Thus, in the density operator formalism, the density operator ρ is the state of the system just like $|Q\rangle$ is the state in the state vector formalism. Instead of state vector, we now talk about *state operator*.

As we said in the introduction to this section, the real power of the density operator formalism is best shown when we are confronted with complex quantum systems. However, simple situations can also be understood using this framework. For example, we may be interested in the mathematical description of an individual qubit in a Bell pair or the effect of the environment on a single qubit. In both cases, we need to describe the individual qubits in the language of the density operator.

The state operator of a single qubit, in general, is given by

$$\rho_q = \sum_{s=1}^{2} p_s|O_s\rangle\langle O_s|\,, \tag{2.25}$$

where $\{|O_1\rangle, |O_2\rangle\}$ is a basis for $\mathcal{H}_O$. For instance, the observational states can be the basis states $\{|0\rangle, |1\rangle\}$ or $\{|+\rangle, |-\rangle\}$. Of course, in general, $\{|O_1\rangle, |O_2\rangle\}$ does not need to be an orthonormal basis.

The elements of the *density matrix* of a single qubit are then given by

$$[\rho_q]_{s's''} = \langle O_{s'}|\rho_q|O_{s''}\rangle = \sum_s p_s\langle O_{s'}|O_s\rangle\langle O_s|O_{s''}\rangle\,. \tag{2.26}$$

Let see how this applies to a concrete example.

Suppose there is a source of single qubits and the measurements have been made along the computational basis vectors $|0\rangle$ and $|1\rangle$. Denoting the probability distribution by $\{|0\rangle, p_0\,; |1\rangle, p_1 = 1 - p_0\}$, we have that the state operator is

$$\rho_q = p_0|0\rangle\langle 0| + (1 - p_0)|1\rangle\langle 1|\,. \tag{2.27}$$

Representing the basis vectors by the usual column vectors, $|0\rangle = [1\ 0]^T$ and $|1\rangle = [0\ 1]^T$, the corresponding projectors are

$$\Pi_{|0\rangle} = |0\rangle\langle 0| = \begin{bmatrix} 1 \\ 0 \end{bmatrix} \begin{bmatrix} 1 & 0 \end{bmatrix} = \begin{bmatrix} 1 & 0 \\ 0 & 0 \end{bmatrix}\,, \tag{2.28}$$

$$\Pi_{|1\rangle} = |1\rangle\langle 1| = \begin{bmatrix} 0 \\ 1 \end{bmatrix} \begin{bmatrix} 0 & 1 \end{bmatrix} = \begin{bmatrix} 0 & 0 \\ 0 & 1 \end{bmatrix}\,. \tag{2.29}$$

The density matrix for the qubit is, then,

$$\rho_q = \begin{bmatrix} p_0 & 0 \\ 0 & 1 - p_0 \end{bmatrix}\,. \tag{2.30}$$

Alternatively, the elements of the density matrix in the computational basis $\{|0\rangle, |1\rangle\}$ for a single qubit can be found by using (2.26),

$$[\rho_q]_{ij} = \langle i|\rho_q|j\rangle = \sum_k p_k \langle i|k\rangle\langle k|j\rangle\,, \tag{2.31}$$

that is,

$$\rho_q = \begin{bmatrix} \langle 0|\rho_q|0\rangle & \langle 0|\rho_q|1\rangle \\ \langle 1|\rho_q|0\rangle & \langle 1|\rho_q|1\rangle \end{bmatrix}\,, \tag{2.32}$$

which, of course, is equal to (2.30).

There is a geometric representation of single-qubit states that sometimes is useful (especially for quantum computer programs). Since any 2×2 complex matrix can always be written as a (real) linear combination of the identity and the Pauli matrices, we can write

$$\rho_q = c_I I + \sum_a c_a \sigma_a = \frac{1}{2}\, I + \frac{1}{2} \sum_a B_a \sigma_a\,. \tag{2.33}$$

In the last expression we have simply chosen $c_I = 1/2$ and wrote $c_a = B_a/2$.

Exercise 2.4. Show that ρ_q satisfies all the properties of a density operator.

More conveniently, we can write

$$\rho_q = \frac{1}{2}\big(I + \mathbf{B}\cdot\boldsymbol{\sigma}\big)\,. \tag{2.34}$$

The real vector $\mathbf{B} \in \mathbb{R}^3$ is known as the *Bloch vector*. We indicate the dependence on this vector by writing $\rho_q = \rho_q\big(\mathbf{B}\big)$. In Cartesian coordinates,

$$\rho_q = \rho_q\big(\mathbf{B} = [B_x\ B_y\ B_z]^T\big)\,. \tag{2.35}$$

Exercise 2.5. Write (2.34) explicitly in terms of the components of the Bloch vector.

Exercise 2.6. Show that the density matrix ρ_q of Exercise 2.5 has eigenvalues $(1 \pm \|\mathbf{B}\|)/2$.

Since the eigenvalues of a density operator are always equal to or greater than zero, we have that $\|\mathbf{B}\| \leq 1$. Thus, geometrically speaking, the state of a single qubit can be represented by a vector inside or on the boundary of a closed unit ball. This unit ball is called the *Bloch ball*. The Bloch ball differs from the Bloch sphere because the former includes the points inside the closed unit ball, $\|\mathbf{B}\| < 1$, whereas the latter consists only of points on the surface, $\|\mathbf{B}\| = 1$. Points on the Bloch sphere represent single-qubit pure states.

Since points on a sphere can be parameterized by the spherical angles (θ, ϕ), there is a one-to-one correspondence between single-qubit state vector $|q(\theta, \phi)\rangle$ and unit Bloch vectors. For example, for $\mathbf{B} = [0\ 0\ 1]^T$,

$$\rho_q\big(\mathbf{B} = [0\ 0\ 1]^T\big) = \frac{1}{2}(I + \sigma_z) = \begin{bmatrix} 1 & 0 \\ 0 & 0 \end{bmatrix} = \begin{bmatrix} 1 & 0 \end{bmatrix} \begin{bmatrix} 1 \\ 0 \end{bmatrix} = |0\rangle\langle 0| \,. \tag{2.36}$$

Thus, we conclude that there is a correspondence

$$\mathbf{B} = [0\ 0\ 1]^T \longleftrightarrow |q(\theta = 0, \phi)\rangle = |0\rangle \,. \tag{2.37}$$

Exercise 2.7. Show that

$$\mathbf{B} = [1\ 0\ 0]^T \longleftrightarrow |q(\theta = \pi/2, \phi = 0)\rangle = |+\rangle \,, \tag{2.38}$$

$$\mathbf{B} = [0\ 1\ 0]^T \longleftrightarrow |q(\theta = \pi/2, \phi = \pi/2)\rangle = \big(|0\rangle + i|1\rangle\big)/\sqrt{2} \,. \tag{2.39}$$

Exercise 2.8. What are the Bloch vectors corresponding to $|1\rangle$ and $|-\rangle$?

Finally, note that, if $\mathbf{B} = [0\ 0\ 0]^T$, then

$$\begin{aligned} \rho_q\big(\mathbf{B} = [0\ 0\ 0]^T\big) = \frac{1}{2}\, I &= \frac{1}{2} \begin{bmatrix} 1 & 0 \\ 0 & 0 \end{bmatrix} + \frac{1}{2} \begin{bmatrix} 0 & 0 \\ 0 & 1 \end{bmatrix} \\ &= \frac{1}{2}\, |0\rangle\langle 0| + \frac{1}{2}\, |1\rangle\langle 1| \,. \end{aligned} \tag{2.40}$$

This means that the probability distribution is $\{|0\rangle\,, 1/2; |1\rangle\,, 1/2\}$. In other words, at the center of the Bloch ball is the mixed state with equal probability of being measured at $|0\rangle$ or $|1\rangle$.

2.2 Multipartite Systems

A *bipartite system* is a composite quantum system consisting of two physical subsystems. In the state vector formalism, we describe it by a vector in the tensor product Hilbert space $\mathcal{H} = \mathcal{H}_Q \otimes \mathcal{H}_{Q'}$, where $\mathcal{H}_Q$ and $\mathcal{H}_{Q'}$ are the Hilbert spaces of the subsystems Q and Q', respectively. For example, the state of a two-qubit system is described by a vector $|q_2\rangle = |q\rangle_{qq'} \in \mathcal{H}_{q_2} = \mathcal{H}_q \otimes \mathcal{H}_{q'}$. A *multipartite system* is a composite quantum system made of multiple physical subsystems. For example, an n qubit, with $n \geq 2$, is a multipartite system. Atoms and molecules, as well as the whole universe, are multipartite systems. From the physical point to view, the most salient feature of these systems is that they are highly entangled. Our goal here is to understand how the density operator formalism can be used to describe these

types of systems and, most importantly, their individual parts. Instead of explaining the most general case, however, we will illustrate the procedure by presenting the already familiar case of the two-qubit system.

In the usual state vector formalism, the state of a two-qubit system, ideally isolated from the rest of the universe, can be written as a linear superposition of the computational basis vectors $\{|i\ i'\rangle\}$ of $\mathcal{H}_{q_2}$, where $i, i' = 0, 1$,

$$|q_2\rangle = \sum_{i,i'} \alpha_{ii'} |i\ i'\rangle \,. \tag{2.41}$$

(see equation (2.14) of QC1). The density operator of this pure state is given by

$$\begin{aligned} \rho_{q_2} = \rho_{qq'} = |q_2\rangle\langle q_2| &= \Big(\sum_{i,i'} \alpha_{ii'} |i\ i'\rangle \Big) \Big(\sum_{j,j'} \alpha^*_{jj'} \langle j\ j'| \Big) \\ &= \sum_{i,j,i',j'} \alpha_{ii'} \alpha^*_{jj'} |i\ i'\rangle\langle j\ j'| \,. \end{aligned} \tag{2.42}$$

Exercise 2.9. Show that

$$\alpha_{ii'} \alpha^*_{jj'} = \langle i\ i'|\rho_{qq'}|j\ j'\rangle \,. \tag{2.43}$$

Using this result and defining the matrix elements

$$[\rho_{qq'}]_{ii',\,jj'} = \alpha_{ii'} \alpha^*_{jj'} = \langle i\ i'|\rho_{qq'}|j\ j'\rangle \,, \tag{2.44}$$

we arrive at

$$\rho_{qq'} = \sum_{i,j,i',j'} [\rho_{qq'}]_{ii',\,jj'} |i\ i'\rangle\langle j\ j'| \,. \tag{2.45}$$

Its matrix representation in the computational basis of $\mathcal{H}_{q_2}$ is,

$$\rho_{qq'} = \begin{bmatrix} \langle 0\ 0|\rho_{qq'}|0\ 0\rangle & \langle 0\ 0|\rho_{qq'}|0\ 1\rangle & \langle 0\ 0|\rho_{qq'}|1\ 0\rangle & \langle 0\ 0|\rho_{qq'}|1\ 1\rangle \\ \langle 0\ 1|\rho_{qq'}|0\ 0\rangle & \langle 0\ 1|\rho_{qq'}|0\ 1\rangle & \langle 0\ 1|\rho_{qq'}|1\ 0\rangle & \langle 0\ 1|\rho_{qq'}|1\ 1\rangle \\ \langle 1\ 0|\rho_{qq'}|0\ 0\rangle & \langle 1\ 0|\rho_{qq'}|0\ 1\rangle & \langle 1\ 0|\rho_{qq'}|1\ 0\rangle & \langle 1\ 0|\rho_{qq'}|1\ 1\rangle \\ \langle 1\ 1|\rho_{qq'}|0\ 0\rangle & \langle 1\ 1|\rho_{qq'}|0\ 1\rangle & \langle 1\ 1|\rho_{qq'}|1\ 0\rangle & \langle 1\ 1|\rho_{qq'}|1\ 1\rangle \end{bmatrix} . \tag{2.46}$$

The density operator formalism states that the density operator of a quantum subsystem is obtained by taking the *partial trace* of the density operator of the entire system. They are known as *reduced density operators*. For example, the state operator of the first qubit is the reduced density operator of the two-qubit system (2.42),

$$\rho_q = \mathrm{Tr}_{q'} \rho_{qq'} = \sum_{i'} \langle \cdot\ i'|\rho_{qq'}|\cdot\ i'\rangle \,. \tag{2.47}$$

The dots in the first positions indicate that we leave untouched the first qubit and trace out over the second Hilbert space. Similarly, the density operator description of the second qubit is,

$$\rho_{q'} = \mathrm{Tr}_q \rho_{qq'} = \sum_{i} \langle\, i\ \cdot |\rho_{qq'}|\, i\ \cdot\rangle \,. \tag{2.48}$$

The dots in the second positions now indicate that we leave untouched the second qubit and trace out over the first Hilbert space.

The matrix elements of the reduced density matrix of the first qubit, in the computational basis of $\mathcal{H}_q$, are given by

$$\begin{aligned}[\rho_q]_{ij} &= \langle\, i\ \cdot |\rho_q|\, j\ \cdot \rangle = \langle\, i\ \cdot |\sum_{i'} \langle \cdot\ i'|\rho_{qq'}|\cdot\ i'\rangle |\, j\ \cdot \rangle \\ &= \sum_{i'} \langle i\ i'|\rho_{qq'}|j\ i'\rangle = \sum_{i'} [\rho_{qq'}]_{ii',ji'}\,. \end{aligned} \tag{2.49}$$

That is,

$$[\rho_q]_{ij} = \langle i\ 0|\rho_{qq'}|j\ 0\rangle + \langle i\ 1|\rho_{qq'}|j\ 1\rangle = [\rho_{qq'}]_{i0,j0} + [\rho_{qq'}]_{i1,j1}\,, \tag{2.50}$$

or, more explicitly,

$$\rho_q = \frac{1}{2}\begin{bmatrix} \langle 0\ 0|\rho_{qq'}|0\ 0\rangle + \langle 0\ 1|\rho_{qq'}|0\ 1\rangle & \langle 0\ 0|\rho_{qq'}|1\ 0\rangle + \langle 0\ 1|\rho_{qq'}|1\ 1\rangle \\ \langle 1\ 0|\rho_{qq'}|0\ 0\rangle + \langle 1\ 1|\rho_{qq'}|0\ 1\rangle & \langle 1\ 0|\rho_{qq'}|1\ 0\rangle + \langle 1\ 1|\rho_{qq'}|1\ 1\rangle \end{bmatrix}. \tag{2.51}$$

Thus, given the matrix density of a two-qubit system (2.46), it suffices to look at this matrix and add the appropriate elements to find the reduced density matrix ρ_q.

Exercise 2.10. Find the matrix elements of the reduced density matrix of the second qubit.

As an example, consider again the Bell state $|\beta_0\rangle$ given in (2.1). Its density operator, in the computational basis of $\mathcal{H}_{q_2}$, is

$$\begin{aligned}\rho_{\beta_0} &= |\beta_0\rangle\langle\beta_0| \\ &= \frac{1}{2}\big(|0\rangle|0\rangle\langle 0|\langle 0| + |0\rangle|0\rangle\langle 1|\langle 1| + |1\rangle|1\rangle\langle 0|\langle 0| + |1\rangle|1\rangle\langle 1|\langle 1|\big)\,. \end{aligned} \tag{2.52}$$

Recall that in our notation, $|i\rangle|i'\rangle = |i\ i'\rangle$ and $\langle j|\langle j'| = \langle j\ j'|$. Thus, the reduced density matrix describing the second qubit is,

$$\begin{aligned}\rho_{q'} = \mathrm{Tr}_q \rho_{\beta_0} &= \sum_{i=0,1} \langle i|\langle\, \cdot\, |\rho_{\beta_0}|i\rangle|\, \cdot\,\rangle \\ &= \langle 0|\langle\, \cdot\, |\rho_{\beta_0}|0\rangle|\, \cdot\,\rangle + \langle 1|\langle\, \cdot\, |\rho_{\beta_0}|1\rangle|\, \cdot\,\rangle\,. \end{aligned} \tag{2.53}$$

The explicit calculation gives,

$$\begin{aligned}\rho_{q'} &= \frac{1}{2}\big(\langle 0|0\rangle|0\rangle\langle 0|0\rangle\langle 0| + \langle 0|0\rangle|0\rangle\langle 1|0\rangle\langle 1| + \langle 0|1\rangle|1\rangle\langle 0|0\rangle\langle 0| + \langle 0|1\rangle|1\rangle\langle 1|0\rangle\langle 1|\big) \\ &+ \frac{1}{2}\big(\langle 1|0\rangle|0\rangle\langle 0|1\rangle\langle 0| + \langle 1|0\rangle|0\rangle\langle 1|1\rangle\langle 1| + \langle 1|1\rangle|1\rangle\langle 0|1\rangle\langle 0| + \langle 1|1\rangle|1\rangle\langle 1|1\rangle\langle 1|\big) \\ &= \frac{1}{2}\big(|0\rangle\langle 0| + |1\rangle\langle 1|\big)\,. \end{aligned} \tag{2.54}$$

Do not forget that this is an operator on the second qubit, $\rho_{q'}\colon \mathcal{H}_{q'} \mapsto \mathcal{H}_{q'}$. In matrix form,

$$\rho_{q'} = \frac{1}{2}\begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}. \tag{2.55}$$

Similarly, it is straightforward to find that

$$\rho_q = \frac{1}{2}\left(|0\rangle\langle 0| + |1\rangle\langle 1|\right) = \frac{1}{2}\begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}, \tag{2.56}$$

where $\rho_q \colon \mathcal{H}_q \mapsto \mathcal{H}_q$.

Exercise 2.11. Show that,

$$\rho_{\beta_0} = \frac{1}{2}\begin{bmatrix} 1 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 1 & 0 & 0 & 1 \end{bmatrix}. \tag{2.57}$$

From here, how do you get the reduced density matrices (2.55) and (2.56)?

Exercise 2.12. Repeat these calculations for the other Bell states (see QC1, Subsection 4.3).

More generally, we could have considered a bipartite system QQ' made of two subsystems Q and Q'. In this case, we denote by $\{|I\rangle\}$ the orthonormal basis of the Hilbert space $\mathcal{H}_Q$, with $I = 1, 2, \ldots, \dim \mathcal{H}_Q$, and by $\{|I'\rangle\}$ that of $\mathcal{H}_{Q'}$, with $I' = 1, 2, \ldots, \dim \mathcal{H}_{Q'}$. Any observational state $|\Psi_s\rangle$, with $s = 1, 2, \ldots, \dim \mathcal{H}_Q \times \dim \mathcal{H}_{Q'}$, will be described by a state vector

$$|\Psi_s\rangle = \sum_{I=1}^{\dim \mathcal{H}_Q} \sum_{I'=1}^{\dim \mathcal{H}_{Q'}} \alpha_{s,II'} |I\rangle|I'\rangle = \sum_{I,I'} \alpha_{s,II'} |I\, I'\rangle\,. \tag{2.58}$$

The state operator of the bipartite system is then

$$\begin{aligned} \rho_{QQ'} = \sum_s p_s |\Psi_s\rangle\langle\Psi_s| &= \sum_s p_s \sum_{I,I'} \alpha_{s,II'} |I\, I'\rangle \sum_{J,J'} \alpha^*_{s,JJ'} \langle J\, J'| \\ &= \sum_{I,J,I',J'} \Big(\sum_s p_s\, \alpha_{s,II'} \alpha^*_{s,JJ'} \Big) |I\rangle\langle J| \otimes |I'\rangle\langle J'| \\ &= \sum_{I,J,I',J'} [\rho_{QQ'}]_{II',JJ'} |I\rangle\langle J| \otimes |I'\rangle\langle J'|\,, \end{aligned} \tag{2.59}$$

where we have defined the matrix elements

$$[\rho_{QQ'}]_{II',JJ'} = \sum_s p_s\, \alpha_{s,II'} \alpha^*_{s,JJ'}\,. \tag{2.60}$$

Exercise 2.13. What are the conditions on $[\rho_{QQ'}]_{II',JJ'}$ if $\rho_{QQ'}$ is the density operator of a separable system?

Tracing out over the Hilbert space of the subsystem Q', we get

$$\begin{aligned} \rho_Q = \mathrm{Tr}_{Q'} \rho_{QQ'} &= \mathrm{Tr}_{Q'} \Big[\sum_{I,J,I',J'} [\rho_{QQ'}]_{II',JJ'} |I\rangle\langle J| \otimes |I'\rangle\langle J'| \Big] \\ &= \sum_{I,J,I',J',K'} [\rho_{QQ'}]_{II',JJ'} |I\rangle\langle J| \otimes \langle K'|I'\rangle\langle J'|K'\rangle\,. \end{aligned} \tag{2.61}$$

Using that $\delta_{K'I'}\delta_{J'K'} = \delta_{I'J'}$, we arrive at the reduced density operator of subsystem Q',

$$\rho_Q = \sum_{I,J,I'} [\rho_{QQ'}]_{II',JI'} |I\rangle\langle J| \,. \tag{2.62}$$

Exercise 2.14. Compute explicitly the reduced density operator of the second subsystem.

2.3 State Evolution

The true power of the density operator formalism shows when studying subsystems of highly entangled multipartite systems. In particular, as we will see next, it provides a neat mathematical description of the evolution of the individual subsystems of a bipartite system. Again, let us start recalling how the state vector formalism approaches the problem.

Suppose we know the state vector of an isolated quantum system at some initial time t_0, call it $|Q(t_0)\rangle$. According to the postulates of conventional quantum mechanics, the state of the system at any other time t is obtained by applying the unitary time evolution operator $U(t, t_0)$,

$$|Q(t_0)\rangle \longmapsto |Q(t)\rangle = U(t, t_0)|Q(t_0)\rangle \,. \tag{2.63}$$

If we assume that the system stays isolated, the time evolution operator takes the simple form $U(t, t_0) = \exp(-i\hat{H}(t - t_0))$, where $\hat{H}$ is the time-independent Hamiltonian operator. Note that, the time evolution operator $U(t, t_0)$ acts on the Hilbert space $\mathcal{H}_{t_0}$ of the system at time t_0 and gives a vector in the Hilbert space $\mathcal{H}_t$ at time t,

$$U(t, t_0) \colon \mathcal{H}_{t_0} \to \mathcal{H}_t \,, \qquad |Q(t_0)\rangle \mapsto |Q(t)\rangle = U(t, t_0)|Q(t_0)\rangle \,. \tag{2.64}$$

This is practically all we need to know, at least theoretically, to predict the time evolution of a closed system. In the density operator formalism, the basic principles are also easy to formulate.

Because the system is initially in the state vector $|Q(t_0)\rangle$, the density operator is simply the pure state

$$\rho(t_0) = |Q(t_0)\rangle\langle Q(t_0)| \,. \tag{2.65}$$

Using the formula (2.63) for the time evolution, the corresponding density operator at time t can be found by simply saying that

$$\begin{aligned} \rho(t) = |Q(t)\rangle\langle Q(t)| &= U(t, t_0)|Q(t_0)\rangle\langle Q(t_0)|U^\dagger(t, t_0) \\ &= U(t, t_0)\rho(t_0)U^\dagger(t, t_0) \,. \end{aligned} \tag{2.66}$$

Note that, while in the standard formalism of quantum mechanics the initial and final state vectors are related by the time evolution operator (2.64), in the density operator formalism, the initial and final states are related by a new kind of operator that acts on the space of density operators at time t_0 and gives a density operator at time t,

$$\Phi(t, t_0) \colon \mathcal{D}(t_0) \to \mathcal{D}(t) \,, \qquad \rho(t_0) \mapsto \rho(t) = \Phi(t, t_0)\rho(t_0) \,. \tag{2.67}$$

Operators such as this, that connect two operator spaces, are called *superoperators* by mathematicians. Physicists prefer to call them *quantum maps* or *quantum channels*.

Exercise 2.15. If the initial quantum system is in a mixed state, how does it evolve in time? Assume that $P(|Q_s\rangle) = P(U|Q_s\rangle)$.

The density operator formalism applies the same idea to subsystems of multipartite quantum systems. That is, if QQ' is a bipartite system and $\rho_Q(t_0)$ is the reduced density operator of the subsystem Q, the formalism states that there is a superoperator

$$\Phi_Q(t,t_0)\colon \mathcal{D}_Q(t_0) \to \mathcal{D}_Q(t)\,, \qquad \rho_Q(t_0) \mapsto \rho_Q(t) = \Phi_Q(t,t_0)\rho_Q(t_0)\,. \tag{2.68}$$

Exercise 2.16. Prove this assertion.

Let us see this in more detail. Imagine two quantum systems Q and Q' that are completely unentangled from each other as well as from the rest of the universe. Since they are unentangled, the density operator that describes the bipartite system at time t_0 is simply

$$\rho_{QQ'}(t_0) = \rho_Q(t_0) \otimes \rho_{Q'}(t_0)\,. \tag{2.69}$$

Suppose, moreover, that we know nothing about Q but have a statistical knowledge of subsystem Q'. For example, we know that at time t_0, Q' has a probability distribution $\{|Q'_0\rangle, p_{Q'_0}\}$, where $\{|Q'_0\rangle\}$ is a basis for the Hilbert space $(\mathcal{H}_{Q'})_0$ of the system Q' at time t_0. Thus, the state operator of Q' at time t_0 is

$$\rho_{Q'}(t_0) = \sum_{Q'_0} p_{Q'_0} |Q'_0\rangle\langle Q'_0|\,, \tag{2.70}$$

The state operator of the bipartite system is then

$$\rho_{QQ'}(t_0) = \rho_Q(t_0) \sum_{Q'_0} p_{Q'_0} |Q'_0\rangle\langle Q'_0|\,, \tag{2.71}$$

where, for simplicity, we have omitted the tensor product symbol.

Now, suppose that after time t_0, Q and Q' interact for a time period $\Delta t = t - t_0$. We still know nothing about Q, of course; however, we do know that, since there is no interaction with the rest of the universe, the bipartite system QQ' evolves unitarily,

$$\rho_{QQ'}(t_0) \mapsto \rho_{QQ'}(t) = U(t,t_0)\rho_{QQ'}(t_0)U^\dagger(t,t_0)\,, \tag{2.72}$$

where

$$U(t,t_0)\colon (\mathcal{H}_Q)_0 \otimes (\mathcal{H}_{Q'})_0 \to (\mathcal{H}_Q)_t \otimes (\mathcal{H}_{Q'})_t\,. \tag{2.73}$$

With only this information, we want to know how the subsystem Q evolves in time.

The state operator of subsystem Q at time t is obtained as follows by the formula of the reduced density operator given above,

$$\rho_Q(t) = \mathrm{Tr}_{Q'}\rho_{QQ'}(t) = \sum_{Q'_t} \langle\, \cdot\, Q'_t\, |\rho_{QQ'}(t)|\, \cdot\, Q'_t\,\rangle\,. \tag{2.74}$$

The use of the dots is to emphasize that this is an operator and not a real number.

Keeping this in mind, we now drop the dots and get,

$$\begin{aligned}\rho_Q(t) &= \sum_{Q'_t}\langle Q'_t|\rho_{QQ'}(t)|Q'_t\rangle \\ &= \sum_{Q'_t}\langle Q'_t|U(t,t_0)\rho_{QQ'}(t_0)U^\dagger(t,t_0)|Q'_t\rangle \\ &= \sum_{Q'_t}\langle Q'_t|U(t,t_0)\left(\rho_Q(t_0)\sum_{Q'_0}p_{Q'_0}|Q'_0\rangle\langle Q'_0|\right)|U^\dagger(t,t_0)|Q'_t\rangle \\ &= \sum_{Q'_0,Q'_t}\sqrt{p_{Q'_0}}\,\langle Q'_t|U(t,t_0)|Q'_0\rangle\rho_Q(t_0)\sqrt{p_{Q'_0}}\,\langle Q'_0|U^\dagger(t,t_0)|Q'_t\rangle\,. \end{aligned} \tag{2.75}$$

It is common to write this expression in terms of the *Kraus operators*,

$$K_{Q'_tQ'_0} = \sqrt{p_{Q'_0}}\langle Q'_t|U(t,t_0)|Q'_0\rangle\,, \tag{2.76}$$

with,

$$K_{Q'_tQ'_0}\colon (\mathcal{H}_Q)_0 \to (\mathcal{H}_Q)_t\,. \tag{2.77}$$

Exercise 2.17. Show that the Kraus operators satisfy the completeness relation,

$$\sum_{Q'_0,Q'_t} K^\dagger_{Q'_tQ'_0}K_{Q'_tQ'_0} = 1\,. \tag{2.78}$$

Then, the density operator of subsystem Q at time t is,

$$\rho_Q(t) = \sum_{Q'_0,Q'_t} K_{Q'_tQ'_0}\rho_Q(t_0)K^\dagger_{Q'_tQ'_0}\,. \tag{2.79}$$

This expression is known as the *Kraus representation* of the density operator of subsystem Q.

Exercise 2.18. Show that $\rho_Q(t)$ satisfies all the properties of a density operator.

In conclusion, the state evolution of a quantum subsystem Q in the density operator formalism is given by the action of a superoperator $\Phi_Q(t,t_0)$, where the latter is defined in terms of the Kraus operators,

$$\begin{aligned}\Phi_Q(t,t_0)\colon \mathcal{D}_Q(t_0)\to\mathcal{D}_Q(t)\,,\quad \rho_Q(t_0)\mapsto\rho_Q(t) \\ &= \Phi_Q(t,t_0)\rho_Q(t_0) \\ &= \sum_{Q'_0,Q'_t} K_{Q'_tQ'_0}\rho_Q(t_0)K^\dagger_{Q'_tQ'_0}\,. \end{aligned} \tag{2.80}$$

Q' indicates everything in the bipartite system not included in Q.

This expression of the superoperator $\Phi_Q(t,t_0)$, allow us to prove some of its main properties. For example, that it is trace preserving,

$$\begin{aligned}\mathrm{Tr}\big[\rho_Q(t)\big] &= \mathrm{Tr}\big[\sum_{Q'_0,Q'_t} K_{Q'_tQ'_0}\rho_Q(t_0)K^\dagger_{Q'_tQ'_0}\big] = \sum_{Q'_0,Q'_t}\mathrm{Tr}\big[K_{Q'_tQ'_0}\rho_Q(t_0)K^\dagger_{Q'_tQ'_0}\big] \\ &= \sum_{Q'_0,Q'_t}\mathrm{Tr}\big[K^\dagger_{Q'_tQ'_0}K_{Q'_tQ'_0}\rho_Q(t_0)\big] = \mathrm{Tr}\big[\sum_{Q'_0,Q'_t}K^\dagger_{Q'_tQ'_0}K_{Q'_tQ'_0}\rho_Q(t_0)\big] \\ &= \mathrm{Tr}\big[(\sum_{Q'_0,Q'_t}K^\dagger_{Q'_tQ'_0}K_{Q'_tQ'_0})\rho_Q(t_0)\big] = \mathrm{Tr}\big[\rho_Q(t_0)\big]\,. \end{aligned} \tag{2.81}$$

It is, as well, easy to show that the superoperator $\Phi_Q(t,t_0)$ is linear, that is, that

$$\Phi_Q(t,t_0)\big(a\rho_Q(t_0)+b\rho'_Q(t_0)\big)=a\Phi_Q(t,t_0)\rho_Q(t_0)+b\Phi_Q(t,t_0)\rho'_Q(t_0)\,. \tag{2.82}$$

Exercise 2.19. Prove the linearity of $\Phi_Q(t,t_0)$.

2.4 Measurement

Now that we know how to describe a quantum system in terms of its density operator, in particular, its evolution in time, we would like to understand how to obtain information about the system. In other words, we want to discuss the measurement process in the density operator formalism. Let us start, though, with a quick review of how measurements are described in the state vector formalism.

As we saw, the state vector formalism, given an n qubit $|Q\rangle=\sum_{s'}\alpha_{s'}|Q_{s'}\rangle$, the probability of measuring a basis state vector $|Q_s\rangle$ is given by the squared modulus of the corresponding coefficient α_s, that is,

$$|Q\rangle=\sum_{s'}\alpha_{s'}|Q_{s'}\rangle\xmapsto{M_s}P(|Q_s\rangle)=p_s=\Big|\langle Q_s|\sum_{s'}\alpha_{s'}|Q_{s'}\rangle\Big|^2=|\alpha_s|^2\,. \tag{2.83}$$

For instance, for a single-qubit state vector $|q\rangle$ expressed in the computational basis $\{|0\rangle\,,|1\rangle\}$, that is, $|q\rangle=\sum_i\alpha_i|i\rangle$, the probability of measuring the state $|i\rangle$ is $|\alpha_i|^2$. Another way of writing this is

$$P(|i\rangle)=|\langle i|q\rangle|^2=\langle i|q\rangle^*\langle i|q\rangle=\langle q|i\rangle\langle i|q\rangle=\langle q|\Pi_i|q\rangle\,, \tag{2.84}$$

where Π_i is the projection operator on the computational basis vector $|i\rangle$, $\Pi_i=|i\rangle\langle i|$. Similarly, the probabilities of measuring the single qubit $|q\rangle$ in the Hadamard basis vectors $|+\rangle$ and $|-\rangle$ are

$$p_\pm=|\langle\pm|q\rangle|^2=\langle\pm|q\rangle^*\langle\pm|q\rangle=\langle q|\pm\rangle\langle\pm|q\rangle=\langle q|P_\pm|q\rangle\,, \tag{2.85}$$

where $\Pi_\pm$ are the projector operators on $|\pm\rangle$, $\Pi_\pm=|\pm\rangle\langle\pm|$. Actually, this argument generalizes to any basis $\{|O_s\rangle\}$ of the Hilbert space $\mathcal{H}_O$ of observational states,

$$p_s=|\langle O_s|q\rangle|^2=\langle O_s|q\rangle^*\langle O_s|q\rangle=\langle q|O_s\rangle\langle O_s|q\rangle=\langle q|\Pi_s|q\rangle\,, \tag{2.86}$$

where $\Pi_s=|O_s\rangle\langle O_s|$. Now, since $\Pi_s=\Pi_s^\dagger$ and $\Pi_s=\Pi_s^2$, we can write

$$p_s=\langle q|\Pi_s|q\rangle=\langle q|\Pi_s^\dagger\Pi_s|q\rangle\,. \tag{2.87}$$

The previous discussion seems to suggest that we can interpret the measurement process M as an operator that maps $|q\rangle\overset{M}{\longmapsto}|O_s\rangle=\Pi_s|q\rangle$. There is, though, a missing normalization factor. In fact, $\Pi_s|q\rangle$ is not a unit vector, the unit vector is $\Pi_s|q\rangle/\sqrt{p_s}$ because

$$\langle O_s|O_s\rangle=\frac{\langle q|\Pi_s^\dagger\Pi_s|q\rangle}{p_s}=1\,. \tag{2.88}$$

From here, we get

$$\langle q|\sum_s\Pi_s^\dagger\Pi_s|q\rangle=\sum_s p_s\,, \tag{2.89}$$

and, since the sum of the probabilities is equal to one and the initial qubit vector is normalized, we must have that

$$\sum_s \Pi_s^\dagger \Pi_s = I\,, \tag{2.90}$$

where I is the identity operator.

A measurement on a single qubit is thus a projector Π_s that transforms

$$|q\rangle \xmapsto{M_s} |O_s\rangle = \frac{\Pi_s|q\rangle}{\sqrt{p_s}} = \frac{\Pi_s|q\rangle}{\sqrt{\langle q|\Pi_s^\dagger \Pi_s|q\rangle}}\,, \tag{2.91}$$

and is subject to the condition (2.90).

Exercise 2.20. Generalize these arguments to an arbitrary n qubit.

More generally, given a basis $\{|O_s\rangle\}$ for the Hilbert space $\mathcal{H}_O$ of observable states, a *measurement operator* is a map

$$M_s\colon \mathcal{H}_Q \to \mathcal{H}_Q\,, \qquad |Q\rangle \mapsto |O_s\rangle = \frac{M_s|Q\rangle}{\sqrt{p_s}}\,, \tag{2.92}$$

that satisfies

$$\sum_s M_s^\dagger M_s = I\,. \tag{2.93}$$

The probability of measuring $|O_s\rangle$ is given by

$$p_s = \langle Q|M_s^\dagger M_s|Q\rangle\,. \tag{2.94}$$

At first, it seems that there is nothing new in these definitions. In fact, we have just seen that projectors meet all these requirements. Note that, however, the definition we just gave is more general because they may be measurement operators M_σ that do not obey the defining properties of projectors ($\Pi_s = \Pi_s^\dagger$ and $\Pi_s = \Pi_s^2$).

Our goal, now, is to translate everything we have just said about measurements to the language of the density operator formalism. That is, given the state operator ρ of a physical system at some initial time, the objective is to find the state operator ρ_s of the system after the measurement has been performed.

Note: In the following, we use some basic probability. In Box 3.1 you can find a quick survey if you want to refresh your memory. We denote join probabilities by $p_{r,s}$ and conditional probabilities by $p_{r|s}$. According to Bayes' theorem (3.7), they are related by $p_{r|s} = p_{r,s}/p_s$.

Suppose that, before the measurement, the quantum system is in the state

$$\rho = \sum_{s'} p_{s'}\ |Q_{s'}\rangle\langle Q_{s'}|\,, \tag{2.95}$$

where $|Q_{s'}\rangle\langle Q_{s'}|\colon \mathcal{H}_O \to \mathcal{H}_O$. If the observational state vectors are $\{|Q_s\rangle\}$, the measurement process is such that

$$\rho \xmapsto{M_s} \rho_s = \sum_{s'} p_{s'|s}\ |Q_{s|s'}\rangle\langle Q_{s|s'}|\,. \tag{2.96}$$

By $p_{s'|s}$, we indicate the probability that the state that enters the apparatus is in $|Q_{s'}\rangle$ given the fact that we measure $|Q_s\rangle$. Using Bayes' theorem,

$$\rho_s = \sum_{s'} \frac{p_{s',\,s}}{p_s}\, |Q_{s|s'}\rangle\langle Q_{s|s'}|\,, \tag{2.97}$$

or, taking into account that $p_{s',\,s} = p_{s,s'}$,

$$\rho_s = \sum_{s'} \frac{p_{s,s'}}{p_s}\, |Q_{s|s'}\rangle\langle Q_{s|s'}|\,. \tag{2.98}$$

Now, since the state $|Q_{s|s'}\rangle$ is the result of a measurement, we can use (2.92) to write,

$$|Q_{s|s'}\rangle = \frac{M_s|Q_{s'}\rangle}{\sqrt{p_{s|s'}}} = \frac{\sqrt{p_{s'}}}{\sqrt{p_{s,s'}}}\, M_s|Q_{s'}\rangle\,, \tag{2.99}$$

giving,

$$|Q_{s|s'}\rangle\langle Q_{s|s'}| = \frac{p_{s'}}{p_{s,s'}}\, M_s|Q_{s'}\rangle\langle Q_{s'}|M_s^\dagger\,. \tag{2.100}$$

Putting all this together,

$$\begin{aligned}
\rho_s &= \sum_{s'} \frac{p_{s'}}{p_s}\, M_s|Q_{s'}\rangle\langle Q_{s'}|M_s^\dagger \\
&= \frac{1}{p_s}\, M_s\Big(\sum_{s'} p_{s'}\, |Q_{s'}\rangle\langle Q_{s'}|\Big)M_s^\dagger \\
&= \frac{M_s\rho M_s^\dagger}{p_s}\,.
\end{aligned} \tag{2.101}$$

Exercise 2.21. Show that for every Hermitian operator A and unit state vector $|Q\rangle$,

$$\langle Q|A|Q\rangle = \mathrm{Tr}\big[A|Q\rangle\langle Q|\big]\,. \tag{2.102}$$

Applying this formula to (2.94), it follows that

$$\begin{aligned}
p_s = \langle Q|M_s^\dagger M_s|Q\rangle &= \mathrm{Tr}\big(M_s^\dagger M_s|Q\rangle\langle Q|\big) \\
&= \mathrm{Tr}(M_s^\dagger M_s\rho)\,.
\end{aligned} \tag{2.103}$$

Inserting this result into (2.101), we conclude that the density operator ρ_s that describes the system after the measurement M_s is related to the initial density operator ρ by the following formula,

$$\rho_s = \frac{M_s\rho M_s^\dagger}{\mathrm{Tr}(M_s^\dagger M_s\rho)}\,. \tag{2.104}$$

3 Information

Given two quantum systems initially unentangled, in the previous section we saw how the operator density formalism describes their individual evolution after they start interacting. Since the interaction produces a bipartite entangled system, it

is natural to ask how much we can "know about" one of the subsystems by only experimenting with the other. This rather simple question raises other interesting questions that, unfortunately, the average physicist is not used to ask him or herself. For instance, what do we exactly mean by "to know about" and how do we measure that "knowledge"? Thankfully, many years ago, computer scientists introduced a concept that precisely answers these questions.

It was Claude Shannon who in the late 1940s discovered that, given a discrete and finite probability distribution $X = \{x_s, P(x_s)\}$, where $s = 1, \ldots, S$, there is a natural way of quantifying the amount of information the system contains. This principle is at the heart of classical information theory. As we will see, the same basic idea applies to a quantum system with discrete and finite probability distribution $\{|\psi_s\rangle, P(|\psi_s\rangle)\}$. Despite the similarities, though, we will see that the two situations also show drastic differences.

Note: The next Box contains an elementary review of probability theory. I suggest you read it because, even if you already know the subject, you will get familiar the notation we will use in the following.

Box 3.1. Elementary probability concepts

Consider any physical experiment with a discrete and finite number of outcomes $\{x_s\}$, with $s = 1, \ldots, s, \ldots S$. Here, for later convenience, we include events with no possibility of occurrence. Suppose, moreover, that we know the probability of occurrence of every event, $P(x_s) = p_s$, subject to the condition $0 \leq p_s \leq 1$. In addition, we impose $\sum_s p_s = 1$. In probability theory, we assume that the probability distribution $\{x_s, p_s\}$ contains everything we can know about the system.

Now, suppose that we perform a different experiment on another system (indeed, the two systems can be equal; however, in order to be as general as possible, let us assume that they are different). As for the first system, suppose that we know all the possible outcomes and their respective probabilities of occurrence, $\{y_r, p_r\}$, with $r = 1, \ldots, r, \ldots, R$. If we analyze the experimental data of both experiments and notice that the probability of simultaneous occurrence of every pair of events (x_s, y_r) is equal to the product of the individual probabilities, that is, if

$$P(x_s, y_r) = P(x_s)P(y_r)\,, \tag{3.1}$$

or, more briefly,

$$p_{s,r} = p_s p_r\,, \tag{3.2}$$

we say that the two systems are *independent*. If, on the contrary, it turns out that

$$p_{s,r} \neq p_s p_r\,, \tag{3.3}$$

even for a single pair (x_s, y_r), we say that the systems are *dependent* or *correlated*. The probability $p_{s,r}$, that is, the probability of simultaneous occurrence of a pair of events, is known as *joint probability*.

Since, for every outcome x_s of the first experiment, there are R different possibilities for the second, it is clear that

$$p_s = p_{s,r=1} + p_{s,r=2} + \ldots + p_{s,r=R} = \sum_r p_{s,r} \,. \tag{3.4}$$

Analogously,

$$p_r = \sum_s p_{r,s} \,. \tag{3.5}$$

These expressions are known as *marginal probabilities*. It is obvious that joint probabilities satisfy

$$p_{s,r} = p_{r,s} \,. \tag{3.6}$$

In addition to the joint probability, we can be interested in the probability of occurrence of an event y_r knowing in advance that x_s has occurred. This is known as *conditional probability*, denoted by $P(y_r|x_s) = p_{r|s}$. *Bayes' theorem* affirms that conditional and joint probabilities are related by the following formula,

$$p_{r|s} = \frac{p_{r,s}}{p_s} \,. \tag{3.7}$$

3.1 Classical Information Theory

In classical communication, the study of the physical and mathematical properties of information is crucial. Specialists in this area, among other things, aim at quantifying the information content of a message so that they can determine how much of it can be transferred and with what degree of efficiency. In addition, the processing and storage of classical information is of key importance for modern computer science. Despite the importance of classical information theory for classical communication and computation, here we will only introduce the classical concepts needed to understand the information theory proper of quantum systems, in particular, open quantum systems.

Our starting point is a discrete and finite probability distribution $X = \{x_s, P(x_s) = p_s\}$, with $s = 1, \ldots, S$. Of course, $0 \leq p_s \leq 1$ and $\sum_s p_s = 1$. Notice that we are including events x_s for which $p_s = 0$, that is, events that have no possibility to occur. From the physical point of view, the probability distribution X can be the list of all possible outcomes x_s of certain experiment and the corresponding probabilities p_s. For instance, the results of tossing an unfair die or the classical states of the particles emanating from a source.

The most basic concept in information theory is that of the *information content of a single event* (sometimes also called the *Shannon entropy of a single event*). It is defined by the following formula,

$$h(x_s) = h_s = \log_2 \frac{1}{p_s} \,. \tag{3.8}$$

The unit of information content is the *bit* (not to be confused with the binary digit $b \in \{0, 1\}$). In these notes, we will stick to the convention used by computer

scientists and write $\log_2$ simply as log. A simpler way of writing the information content of a single event is then

$$h_s = -\log p_s \,. \tag{3.9}$$

For the moment, we are excluding events that cannot occur, that is, $p_s \neq 0$.

The following example will clarify why the information content, h_s, and the probability of occurrence, p_s, are inversely proportional to each other. Imagine that you have a bag full of balls of different colors. Suppose there are N_w white balls and $N_{\bar{w}}$ of another color, with $N_w >> N_{\bar{w}}$. If, on your first trial you pick a white ball, from the information point of view you have not learned much about the system because most probably (after replacing the ball) on your second trial you will pick another white ball. If, on the contrary, on your first trial you pick a ball which is not white, you have a better knowledge of the content of the bag because most probably your second ball will be white. This illustrates why the information content of a single event is interpreted as the degree of uncertainty or surprise. Thus, the more we know about a system, the less information it stores. This is why there no information associated to a system if we know beforehand the experimental result, $-\log 1 = 0$. Finally, since $h_s \geq 0$, it follows that a system with more than one possible outcome will necessarily have $h_s > 0$ for every x_s in X.

Exercise 3.1. Elaborate on the previous example by using specific numbers for the number of balls.

Now, suppose you repeat N times the same experiment/observation on a physical system and the event x_s, for every $s = 1, \ldots, S$, is obtained n_s times. In this case, you may include events that cannot occur, that is, events with $p_s = 0$. Your experimental results are then characterized by the probability distribution $X = \{x_s, n_s/N\}$, where $n_1 + \ldots + n_s = N$. According to Shannon's classical theory, the total information you gain about the system by making these observations is the sum of the individual information contents carried by each event, that is, $n_1 h_1 + \ldots + n_S h_S$. On average, then, each event will carry an information in bits given by

$$\frac{n_1 h_1 + \ldots + n_S h_S}{N} = \frac{n_1}{N}\, h_1 + \ldots + \frac{n_S}{N}\, h_S \,. \tag{3.10}$$

When the number of experiments becomes infinitely large, $N \to \infty$, the average information content per event becomes

$$\begin{aligned} \lim_{N\to\infty} \frac{n_1}{N}\, h_1 + \ldots + \lim_{N\to\infty} \frac{n_S}{N}\, h_S &= p_1 h_1 + \ldots + p_S h_S \\ &= -p_1 \log p_1 - \ldots - p_S \log p_S \,. \end{aligned} \tag{3.11}$$

This quantity is known as the *Shannon entropy of the probability distribution* X,

$$H(X) = -\sum_s p_s \log p_s \,. \tag{3.12}$$

Note that we can have $p_s = 0$ because $0 \log 0 = 0$.

For example, a two-state system with $X = \{x_1, 1/2;\, x_2, 1/2\}$ contains 1 bit of information because

$$H(X) = \sum_s p_s \log \frac{1}{p_s} = \frac{1}{2}\log 2 + \frac{1}{2}\log 2 = 1 \,. \tag{3.13}$$

Similarly, n such independent systems contain n bits of information. Note that, just like the information content of a single qubit, the Shannon entropy is determined solely by the probabilities of the events and not their experimental values.

Since the probabilities are equal to or greater than zero, it follows that the Shannon entropy

$$H(X) \geq 0\,. \tag{3.14}$$

It vanishes only when there is one and only one possible experimental result, that is, when the probability distribution is $\{x, p_x\}$; any other probability distribution has positive Shannon entropy. In fact, it can be proved that the Shannon entropy ranges from zero to $\log N$,

$$0 \leq H(X) \leq \log N\,, \tag{3.15}$$

where N is the number of experiments. The highest Shannon entropy is obtained when all the possible outcomes have equal probability of occurrence, $p_s = 1/N$. In simple words, our ignorance about the system is the highest when all the events are equally probable. Any other probability distribution contains more information.

Exercise 3.2. Prove (3.15) by using the inequality $\ln 1/x \geq 1 - x$, valid for any positive number x.

Consider the following generalization of the classical binary system presented above. This is the simplest non-trivial case with only two possible experimental results (think, for instance, of a bent coin). Its probability distribution is $X = \{x, p_x = p\,; \bar{x}, p_{\bar{x}} = 1 - p\}$ and the Shannon entropy,

$$H(X) = -p \log p - (1 - p) \log(1 - p)\,. \tag{3.16}$$

Instead of assuming that the entropy is given by the *binary probability distribution* X, we can think of it as a function of a single variable,

$$H\colon [0, 1] \to [0, 1]\,, \qquad p \mapsto H(p)\,, \tag{3.17}$$

where

$$H(p) = -p \log p - (1 - p) \log(1 - p)\,. \tag{3.18}$$

Exercise 3.3. For which value of p reaches $H(p)$ its maximum value? Plot $H(p)$. Interpret your result.

Since the maximum entropy happens when the possible events are equally probable, the binary probability function gives $H(1/2) = 1$. From here we conclude that, a Yes/No problem (that is, a problem that in principle can be solved by a sequence of Yes/No questions), has a total Shannon entropy equal to the number of Yes/No questions we have to ask in order to solve it.

We applied the basic information theory introduced above to individual systems with known probability distributions. But, what happens when we have several systems? In principle, they may be correlated, that is, they may show a non-trivial joint probability distribution. Let us see how Shannon's theory works in this case.

Suppose we are given two physical systems, each with its own probability distribution: $X = \{x_s, P(x_s) = p_s\}$, where $s = 1, \ldots, S$, and $Y = \{y_r, P(y_r) = p_r\}$, where $r = 1, \ldots, R$. Since we are assuming that they can be correlated, we can use

their joint probability $p_{s,r}$ to define the *joint information content* or *Shannon joint entropy of two events*,

$$h_{s,r} = -\log p_{s,r}\,. \tag{3.19}$$

This definition applies whether the two systems are correlated, $p_{s,r} \neq p_s p_r$, or not, $p_{s,r} = p_s p_r$. If they are independent of each other, the joint entropy is simply

$$h_{s,r} = -\log p_{s,r} = -\log(p_s p_r) = h_s + h_r\,. \tag{3.20}$$

This should not come as a surprise. Actually, remember that, in statistical physics the thermodynamic entropy is defined in terms of the log function precisely because it is the only function that guarantees this additive property.

The joint entropy of two systems XY is defined according to the Shannon entropy given above. We start by defining the *Shannon marginal joint entropy* of subsystem X,

$$H(X, y_r) = -\sum_s p_{s,r} \log p_{s,r}\,. \tag{3.21}$$

Exercise 3.4. In physical language, what does the marginal entropy tell us?

From here, we define the *Shannon joint entropy* of the composite system XY as,

$$H(X,Y) = \sum_r H(X, y_r) = -\sum_{s,r} p_{s,r} \log p_{s,r}\,. \tag{3.22}$$

Since the joint entropy is clearly symmetric, $p_{s,r} = p_{r,s}$, it follows that

$$H(X,Y) = H(Y,X)\,. \tag{3.23}$$

The joint entropy is thus the average information content carried by a pair of events, one in X and the other in Y.

Exercise 3.5. Show that

$$H(X,Y) \geq H(X)\,, \qquad H(X,Y) \geq H(Y)\,. \tag{3.24}$$

What do these inequalities mean physically? When do the identities hold?

The conditional probability of two events, $p_{r|s}$, can be used in a similar way to define the *conditional information content* or the *Shannon conditional entropy of two events*,

$$h_{r|s} = -\log p_{r|s}\,. \tag{3.25}$$

This is the information content of the event y_r knowing that x_s had already occurred. Since, by Bayes' theorem, $p_{r|s} = p_{r,s}/p_s$ and, by the symmetry of the joint probability, $p_{r,s} = p_{s,r}$, it follows that $p_{r|s} = p_{s|r}p_r/p_s$, thus $h_{r|s} \neq h_{s|r}$ (unless, of course, $p_r = p_s$)

If we define the *marginal conditional entropy* $H(Y|x_s)$ as the Shannon entropy of system Y after having measured the single event x_s of X by

$$H(Y|x_s) = -\sum_r p_{r|s} \log p_{r|s}\,. \tag{3.26}$$

The *Shannon conditional entropy* is the average information content of the events of Y having a complete knowledge of X,

$$H(Y|X) = \sum_s p_s H(Y|x_s) = -\sum_{s,r} p_s p_{r|s} \log p_{r|s} \,. \tag{3.27}$$

Using Bayes' theorem, $p_s p_{r|s} = p_{s,r}$, the conditional Shannon entropy becomes

$$H(Y|X) = -\sum_{s,r} p_{s,r} \log p_{r|s} \,. \tag{3.28}$$

Note that

$$H(Y|X) \geq 0 \,. \tag{3.29}$$

When X and Y are independent, that is, when $p_{s,r} = p_s p_r$,

$$H(Y|X) = -\sum_{s,r} p_s p_r \log \frac{p_s p_r}{p_s} = -\sum_r p_r \log p_r = H(Y) \,. \tag{3.30}$$

Indeed, it can be proved that more generally,

$$H(Y|X) \leq H(Y) \,. \tag{3.31}$$

Thus, on average, the element of surprise of subsystem Y is lesser when we know in advance how the results of Y are conditioned by those of X. Only when the two subsystems are uncorrelated, do the measurements of X provide no information about Y.

Exercise 3.6. Show the inequality (3.31). Hint: use $\ln x \leq x - 1$, for any positive x.

Exercise 3.7. Show that,

$$H(Y|X) = H(Y, X) - H(X) \,. \tag{3.32}$$

What is this identity telling us? Use plain words.

Exercise 3.8. Show and explain in simple words following inequality

$$H(X, Y) \leq H(X) + H(Y) \,. \tag{3.33}$$

Exercise 3.9. Explain the relation

$$H(Y|X) \leq H(Y) \leq H(Y, X) \,. \tag{3.34}$$

Above we have seen that, to every event x_s we must assign the information content $h_s = -\log p_s$ (for $p_s \neq 0$). If the system X is not correlated to any other system, this is all the information carried by this event. However, if we know in advance that this event is conditioned by the outcome y_r of another system Y, we know something about x_s even before it occurs. Therefore, in this case, the information content is h_s minus the conditional information content of the pair of events $(y_r|x_s)$,

$$h_s - h_{s|r} = -\log p_s + \log p_{s|r} = -\log \frac{p_s p_r}{p_{s,r}} \,. \tag{3.35}$$

On average, the information gained per pair of events (x_s, y_r) is, then,

$$\sum_{s,r} p_{s,r}(h_s - h_{s|r}) = -\sum_{s,r} p_{s,r} \log \frac{p_s p_r}{p_{s,r}} \,. \tag{3.36}$$

This quantity is called the *mutual information* of X and Y,

$$I(X;Y) = -\sum_{s,r} p_{s,r} \log \frac{p_s p_r}{p_{s,r}} \,. \tag{3.37}$$

Using the properties of the log function, the mutual information can be written in several equivalent ways. For example,

$$\begin{aligned} I(X;Y) &= -\sum_{s,r} p_{s,r} \log p_s - \sum_{s,r} p_{s,r} \log p_r + \sum_{s,r} p_{s,r} \log p_{s,r} \\ &= -\sum_{s} p_s \log p_s - \sum_{r} p_r \log p_r + \sum_{s,r} p_{s,r} \log p_{s,r} \\ &= H(X) + H(Y) - H(X,Y) \qquad (3.38) \\ &= H(X) - H(X|Y) \,, \qquad (3.39) \end{aligned}$$

where, in the last step, we used (3.32). Since $H(X,Y) = H(Y,X)$, the mutual information is also symmetric,

$$I(X;Y) = I(Y;X) \,. \tag{3.40}$$

This means that given two statistical ensembles, the average information gained is the same whether we observe X or Y. If they are independent, that is, if $p_{s,r} = p_s p_r$, $I(X;Y) = 0$. So, as expected, there is no mutual information.

Exercise 3.10. Show that $I(X;Y) \geq 0$.

Exercise 3.11. Interpret the following Venn diagram. Provide a physical interpretation.

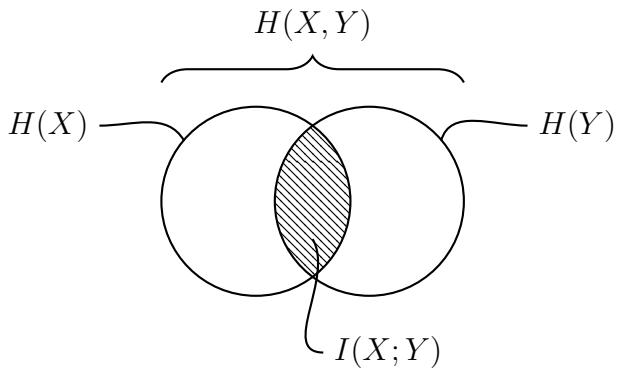

Fig. 2. Venn diagram for classical information concepts.

The following exercises ask you to generalize the definitions and properties given above to more than two systems. To do that, assume that you are given three physical systems X, Y, Z, each with its corresponding probability distribution. In each case, draw the associated Venn diagram.

Exercise 3.12. Show that

$$H(X,Y,Z) = H(X) + H(Y|X) + H(Z|Y,X)\,, \tag{3.41}$$

and

$$H(X,Y,Z) \leq H(X) + H(Y) + H(Z)\,. \tag{3.42}$$

When does the identity hold?

Exercise 3.13. Prove and explain in plain words the following identity,

$$H(X,Y|Z) = H(X|Z) + H(Y|X,Z)\,. \tag{3.43}$$

Exercise 3.14. The *conditional mutual information* is defined as

$$I(X;Y|Z) = H(X|Z) - H(X|Y,Z)\,. \tag{3.44}$$

What is this quantity telling us?

Exercise 3.15. Show that

$$I(X,Y;Z) = I(X,Z) + H(Y;Z|X)\,. \tag{3.45}$$

Exercise 3.16. Prove the *strong subadditivity* relation,

$$I(X;Y|Z) \geq 0\,. \tag{3.46}$$

When does the mutual information vanish?

Exercise 3.17. Generalize the previous results to an arbitrary number of systems $X_1, X_2, \ldots, X_n$.

Exercise 3.18. Justify the following definitions and generalize to n systems,

$$H(X,Y,Z) = -\sum_{s,r,t} p_{s,r,t} \log p_{s,r,t}\,, \tag{3.47}$$

$$H(X,Y|Z) = -\sum_{s,r,t} p_{s,r,t} \log p_{s,r|t}\,, \tag{3.48}$$

$$H(Z|X,Y) = -\sum_{s,r,t} p_{s,r,t} \log p_{t|s,r}\,, \tag{3.49}$$

$$I(X;Y;Z) = -\sum_{s,r,t} p_{s,r,t} \log \frac{p_r p_s p_t}{p_{s,r,t}}\,. \tag{3.50}$$

Use these formulas to confirm your answers of the previous exercises.

Before closing this short introduction to classical information theory, there is a last useful measure we would like to introduce. Imagine you have two macroscopic systems that look exactly the same; for example, a pair of dice or coins. However, after extensive experimentation, you conclude that, in fact, they are not identical. You discover that, under the same experimental conditions, they exhibit different

probability distributions. Let us say that system X is characterized by $\{x_s, p_s\}$ and system $\widetilde{X}$ by $\{x_{\tilde{s}}, p_{\tilde{s}}\}$. The corresponding Shannon entropies are,

$$h(x_s) = -\log p_s\,, \qquad H(X) = -\sum_s p_s \log p_s\,, \tag{3.51}$$

$$h(x_{\tilde{s}}) = -\log p_{\tilde{s}}\,, \qquad H(\widetilde{X}) = -\sum_{\tilde{s}} p_{\tilde{s}} \log p_{\tilde{s}}\,. \tag{3.52}$$

We wish to quantify the dissimilarity of the two probability distributions X and $\widetilde{X}$. For this, we start by introducing the *relative entropy between a pair of events* $(x_s, x_{\tilde{s}}) \in (X, \widetilde{X})$,

$$h(x_s \| x_{\tilde{s}}) = h(x_{\tilde{s}}) - h(x_s) = -\log p_{\tilde{s}} + \log p_s = -\log \frac{p_{\tilde{s}}}{p_s}\,. \tag{3.53}$$

The average relative entropy of a pair of events $(x_s, x_{\tilde{s}})$ per event $x_s \in X$ is what is called the *relative entropy between X and $\widetilde{X}$*,

$$H(X\|\widetilde{X}) = -\sum_{s,\tilde{s}} p_s \log \frac{p_{\tilde{s}}}{p_s} = -\sum_{s,\tilde{s}} p_s (\log p_{\tilde{s}} - \log p_s)\,. \tag{3.54}$$

Relative entropy is also known as *divergence* or *discrimination* (usually denoted with the letter D). Beware that relative entropy is not symmetric,

$$\begin{aligned} H(X\|\widetilde{X}) &= -\sum_{s,\tilde{s}} p_s \log \frac{p_{\tilde{s}}}{p_s} \\ &\neq -\sum_{s,\tilde{s}} p_{\tilde{s}} \log \frac{p_s}{p_{\tilde{s}}} = H(\widetilde{X}\|X)\,. \end{aligned} \tag{3.55}$$

In order to distinguish these two expressions, it is more appropriate to say that $H(X\|\widetilde{X})$ is the entropy of X relative to $\widetilde{X}$, and $H(\widetilde{X}\|X)$ the entropy of $\widetilde{X}$ relative to X.

Exercise 3.19. State in simple words the difference, concerning their information content, between $H(X\|\widetilde{X})$ and $H(\widetilde{X}\|X)$.

Note that, if $X = \widetilde{X}$,

$$H(X\|X) = -\sum_s p_s \log \frac{p_s}{p_s} = 0\,. \tag{3.56}$$

Otherwise, that is, when $X \neq \widetilde{X}$,

$$H(X\|\widetilde{X}) > 0\,. \tag{3.57}$$

Thus, relative entropy is a non-negative quantity,

$$H(X\|\widetilde{X}) \geq 0\,. \tag{3.58}$$

In case one of the systems has only one possible outcome, say $\widetilde{X} = \{x_{\tilde{s}}, p_{\tilde{s}} = 1\}$, then

$$H(X\|x_{\tilde{s}}) = -\sum_s p_s \log \frac{p_{\tilde{s}}}{p_s} = -\sum_s p_s \log \frac{1}{p_s} = H(X)\,. \tag{3.59}$$

Inspired by the definition of relative entropy, we can as well introduce the *joint relative entropy*,

$$H(p_{s,r}\|\tilde{p}_{s,r}) = -\sum_{s,r} p_{s,r} \log \frac{\tilde{p}_{s,r}}{p_{s,r}}\,, \tag{3.60}$$

and the *conditional relative entropy*,

$$H(p_{s|r}\|\tilde{p}_{s|r}) = -\sum_{s,r} p_{s,r} \log \frac{\tilde{p}_{s|r}}{p_{s|r}}\,. \tag{3.61}$$

Exercise 3.20. Show that $H(p_{s|r}\|\tilde{p}_{s|r} = p_s p_r) = I(X,Y)$.

3.2 Quantum Information Theory

In the previous subsection we considered a classical system X as a source of random events characterized by a probability distribution $\{x_s, p_s\}$. Using only this experimental data, in particular, the probabilities, the Shannon entropy defines its information content. For more than two classical systems, their joint and conditional probabilities were used to define other more sophisticated information measures.

Similarly, we know that in the density operator formalism, a quantum system is considered to be a random source of observational states with certain probabilities. It is natural, then, to expect a quantum version of the Shannon entropy as well as of the other information measures in order to quantify the information content in the quantum world. This is what we will do in this subsection.

Now, since classical physics is a special limit of quantum physics, classical information theory is, as a matter of fact, a special case of quantum information theory. This is why there are quantum analogs of the various classical information measures discussed in the previous subsection. However, because of quantum entanglement, we will see that these quantum quantities can differ radically from their classical counterparts.

Given a quantum system Q, whether open or not, its *von Neumann entropy* is defined by

$$S(Q) = S(\rho_Q) = -\mathrm{Tr}_Q\big[\rho_Q \log \rho_Q\big]\,. \tag{3.62}$$

Note that, to distinguish at a glance the classical Shannon entropy from the quantum von Neumann entropy, we have denoted the former by the letter H and the latter by S.

Before going any further, let us show that the quantum von Neumann entropy is, in fact, analogous to the classical Shannon entropy. For this, we need the eigenvalue decomposition of the density operator given in (2.19),

$$\rho_Q = \sum_s p_s |e_s\rangle\langle e_s|\,. \tag{3.63}$$

Inserting this formula in the definition of the von Neumann entropy, gives

$$\begin{aligned} S(\rho_Q) &= -\mathrm{Tr}_Q\big[\rho_Q \log \rho_Q\big] \\ &= -\mathrm{Tr}_Q\big[\sum_s p_s |e_s\rangle\langle e_s| \log \sum_{s'} p_{s'} |e_{s'}\rangle\langle e_{s'}|\big] \\ &= -\mathrm{Tr}_Q\big[\sum_{s,s'} p_s \log p_{s'} |e_s\rangle\langle e_s|e_{s'}\rangle\langle e_{s'}|\big] \\ &= -\mathrm{Tr}_Q\big[\sum_s p_s \log p_s |e_s\rangle\langle e_s|\big] \\ &= -\sum_s p_s \log p_s \mathrm{Tr}\big[|e_s\rangle\langle e_s|\big] \\ &= -\sum_s p_s \log p_s \sum_s p_s \\ &= -\sum_s p_s \log p_s \,. \end{aligned} \tag{3.64}$$

As we wanted to show. Note that, since $0 < p_s < 1$, the von Neumann entropy is semi-positive, $S(\rho_Q) \geq 0$.

It is easy to show that the von Neumann entropy is, in fact, independent of the orthonormal basis chosen for $\mathcal{H}_Q$. Suppose that $\{|e'_s\rangle\}$ is another orthonormal basis for $\mathcal{H}_Q$, related to $\{|e_s\rangle\}$ by $|e_s\rangle = U|e'_s\rangle$, where U is a unitary operator. The density operators are, thus, related by $\rho'_Q = U\rho_Q U^\dagger$. The von Neumann entropy in this new basis is then,

$$\begin{aligned} S(\rho'_Q) &= -\mathrm{Tr}_Q\big[\rho'_Q \log \rho'_Q\big] = -\mathrm{Tr}_Q\big[U\rho_Q U^\dagger \log\big(U\rho_Q U^\dagger\big)\big] \\ &= -\mathrm{Tr}_Q\big[U\rho_Q U^\dagger U(\log \rho_Q)U^\dagger\big] = -\mathrm{Tr}_Q\big[U\rho_Q(\log \rho_Q)U^\dagger\big] \\ &= -\mathrm{Tr}_Q\big[\rho_Q(\log \rho_Q)U^\dagger U\big] = -\mathrm{Tr}_Q\big[\rho_Q \log \rho_Q\big] = S(\rho_Q)\,. \end{aligned} \tag{3.65}$$

Let us see how this works in practice considering the von Neumann entropy of the mixed single-qubit state given in (2.27),

$$\rho_q = p_0|0\rangle\langle 0| + (1-p_0)|1\rangle\langle 1|\,. \tag{3.66}$$

From the definition of the von Neumann entropy,

$$\begin{aligned} S(q) &= -\mathrm{Tr}_q\big[\rho_q \log \rho_q\big] \\ &= -\mathrm{Tr}_q\big[\big(p_0|0\rangle\langle 0| + (1-p_0)|1\rangle\langle 1|\big) \log\big(p_0|0\rangle\langle 0| + (1-p_0)|1\rangle\langle 1|\big)\big] \\ &= -\mathrm{Tr}_q\big[\big(p_0|0\rangle\langle 0| + (1-p_0)|1\rangle\langle 1|\big)\big(\log p_0|0\rangle\langle 0| + \log(1-p_0)|1\rangle\langle 1|\big)\big] \\ &= -\mathrm{Tr}_q\big[\big(p_0|0\rangle\langle 0| + (1-p_0)|1\rangle\langle 1|\big) \log p_0|0\rangle\langle 0|\big] \\ &\quad - \mathrm{Tr}_q\big[\big(p_0|0\rangle\langle 0| + (1-p_0)|1\rangle\langle 1|\big) \log(1-p_0)|1\rangle\langle 1|\big] \\ &= -\mathrm{Tr}_q\big[p_0 \log p_0|0\rangle\langle 0|0\rangle\langle 0|\big] - \mathrm{Tr}_q\big[(1-p_0)\log(1-p_0)|1\rangle\langle 1|1\rangle\langle 1|\big] \\ &= -\mathrm{Tr}_q\big[p_0 \log p_0|0\rangle\langle 0|\big] - \mathrm{Tr}_q\big[(1-p_0)\log(1-p_0)|1\rangle\langle 1|\big] \\ &= -p_0 \log p_0 \langle 0|0\rangle\langle 0|0\rangle - (1-p_0)\log(1-p_0)\langle 1|1\rangle\langle 1|1\rangle \\ &= -p_0 \log p_0 - (1-p_0)\log(1-p_0)\,. \end{aligned} \tag{3.67}$$

We can, of course, calculate this entropy by using an explicit matrix representation of the density operator as the one given in (2.30)), that is,

$$\begin{aligned} S(q) &= -\mathrm{Tr}_q\big[\rho_q \log \rho_q\big] \\ &= -\mathrm{Tr}\left(\begin{bmatrix} p_0 & 0 \\ 0 & 1-p_0 \end{bmatrix}\begin{bmatrix} \log p_0 & 0 \\ 0 & \log(1-p_0) \end{bmatrix}\right) \\ &= -p_0 \log p_0 - (1-p_0)\log(1-p_0)\,. \end{aligned} \tag{3.68}$$

In particular, for $p_0 = p_1 = 1/2$,

$$\rho_q = \frac{1}{2}\,|0\rangle\langle 0| + \frac{1}{2}\,|1\rangle\langle 1|\,, \tag{3.69}$$

and the von Neumann entropy is then

$$S(q) = -\frac{1}{2}\log\frac{1}{2} - \frac{1}{2}\log\frac{1}{2} = -\log\frac{1}{2} = \log 2 = 1\,. \tag{3.70}$$

Note that the von Neumann entropy of the mixed single-qubit state (3.66) is equal to the Shannon entropy (3.16) of a classical binary system. They coincide because the two orthonormal states $|0\rangle$ and $|1\rangle$ are completely distinguishable (no quantum correlation between them), exactly as the outcomes of a binary classical system such as a bent coin.

Exercise 3.21. Show that this result is independent of the matrix representation of the density operator.

For a general pure single-qubit state, the calculation of the entropy is less obvious. First, we use $|q\rangle = \alpha_0\,|0\rangle + \alpha_1\,|1\rangle$ to build the density matrix,

$$\rho_q = |q\rangle\langle q| = \begin{bmatrix} \alpha_0\alpha_0^* & \alpha_0\alpha_1^* \\ \alpha_0^*\alpha_1 & \alpha_1\alpha_1^* \end{bmatrix}\,, \tag{3.71}$$

and then find the eigenvalue decomposition to be used in the formula (3.64).

Exercise 3.22. What are the von Neumann entropies of the pure states $|\pm\rangle\langle\pm|$, where $|\pm\rangle = (|0\rangle \pm |1\rangle)/\sqrt{2}$ are the Hadamard basis states?

Exercise 3.23. What is the von Neumann entropy of any pure single-qubit state $|q\rangle\langle q|$, with $|q\rangle = \alpha_0\,|0\rangle + \alpha_1\,|1\rangle$?

Exercise 3.24. What is the von Neumann entropy of the pure state $|\rho_{\beta_0}\rangle\langle\rho_{\beta_0}|$, where $|\rho_{\beta_0}\rangle$ is the Bell state given in (2.52)?

Exercise 3.25. Show that the von Neumann entropy of any pure state $|Q\rangle\langle Q|$ is equal to zero.

In principle, any quantum system Q can be viewed as part of a greater bipartite system QQ'. The individual subsystem Q, as we know, is described by the reduced density operator $\rho_Q = \mathrm{Tr}_{Q'}\rho_{QQ'}$. It is natural, then, to define the *reduced von Neumann entropy* of Q by

$$\begin{aligned} S(Q) = S(\rho_Q) &= -\mathrm{Tr}_Q\big[\rho_Q \log \rho_Q\big] \\ &= -\mathrm{Tr}_Q\big[\mathrm{Tr}_{Q'}\rho_{QQ'} \log \mathrm{Tr}_{Q'}\rho_{QQ'}\big]\,. \end{aligned} \tag{3.72}$$

Exercise 3.26. Draw the Venn diagram that illustrates this definition.

The *joint von Neumann entropy* is defined by

$$S(Q,Q') = S(\rho_{QQ'}) = -\mathrm{Tr}_{QQ'}\left[\rho_{QQ'}\log\rho_{QQ'}\right]. \tag{3.73}$$

When the two subsystems are decoupled, that is, when $\rho_{QQ'} = \rho_Q\rho_{Q'}$, the von Neumann entropy satisfies the so called *additivity property*,

$$S(\rho_{QQ'}) = S(\rho_Q\rho_{Q'}) = S(\rho_Q) + S(\rho_{Q'}). \tag{3.74}$$

To prove this property, we start with the eigenvalue decomposition of the density operators ρ_Q and $\rho_{Q'}$ and insert them in the definition of the von Neumann entropy,

$$\begin{aligned}
S(\rho_Q\rho_{Q'}) &= -\mathrm{Tr}_{QQ'}\left[(\rho_Q\otimes\rho_{Q'})\log(\rho_Q\otimes\rho_{Q'})\right] \\
&= -\mathrm{Tr}_{QQ'}\left[(\rho_Q\otimes\rho_{Q'})\log(\sum_s p_s|e_s\rangle\langle e_s|\otimes\sum_{s'}p'_{s'}|e'_{s'}\rangle\langle e'_{s'}|)\right] \\
&= -\mathrm{Tr}_{QQ'}\left[(\rho_Q\otimes\rho_{Q'})(\sum_{s,s'}\log(p_sp'_{s'})|e_s\rangle\langle e_s|\otimes|e'_{s'}\rangle\langle e'_{s'}|)\right] \\
&= -\mathrm{Tr}_{QQ'}\left[(\rho_Q\otimes\rho_{Q'})(\sum_s\log p_s|e_s\rangle\langle e_s|\otimes\sum_{s'}|e'_{s'}\rangle\langle e'_{s'}|)\right] \\
&\quad -\mathrm{Tr}_{QQ'}\left[(\rho_Q\otimes\rho_{Q'})(\sum_s|e_s\rangle\langle e_s|\otimes\sum_{s'}\log p'_{s'}|e'_{s'}\rangle\langle e'_{s'}|)\right] \\
&= -\mathrm{Tr}_{QQ'}\left[(\rho_Q\otimes\rho_{Q'})(\log\rho_Q\otimes I_{Q'})\right] - \mathrm{Tr}_{QQ'}\left[(\rho_Q\otimes\rho_{Q'})(I_Q\otimes\log\rho_{Q'})\right] \\
&= -\mathrm{Tr}_{QQ'}\left[\rho_Q\log\rho_Q\otimes\rho_{Q'}\right] - \mathrm{Tr}_{QQ'}\left[\rho_Q\otimes\rho_{Q'}\log\rho_{Q'}\right] \\
&= -\mathrm{Tr}_Q\left[\rho_Q\log\rho_Q\right]\cdot\mathrm{Tr}_{Q'}\rho_{Q'} - \mathrm{Tr}_Q\rho_Q\cdot\mathrm{Tr}_{Q'}\left[\rho_{Q'}\log\rho_{Q'}\right] \\
&= -\mathrm{Tr}_Q\left[\rho_Q\log\rho_Q\right] - \mathrm{Tr}_{Q'}\left[\rho_{Q'}\log\rho_{Q'}\right] \\
&= S(\rho_Q) + S(\rho_{Q'}).
\end{aligned} \tag{3.75}$$

Recall that most generally $\rho_{QQ'} \neq \rho_Q\rho_{Q'}$. It can be proved that in this case, the von Neumann entropy of a bipartite system satisfies the *subadditivity property*,

$$S(\rho_{Q'Q'}) < S(\rho_Q\rho_{Q'}), \tag{3.76}$$

In conclusion,

$$S(Q,Q') \leq S(Q) + S(Q'), \tag{3.77}$$

and the equality holds only when the two subsystems are independent, $\rho_{QQ'} = \rho_Q\rho_{Q'}$.

Exercise 3.27. Prove the subadditivity property of the von Neumann entropy.

Exercise 3.28. If Q and Q' are subsystems of the bigger tripartite system $QQ'Q''$, how would you define the joint entropy $S(Q,Q')$ in terms of the density operator $\rho_{QQ'Q''}$? Draw the corresponding Venn diagram.

Exercise 3.29. Generalize the previous exercise to an arbitrary multipartite system.

In (3.24), we saw that the joint entropy of two classical systems is greater than or equal to the entropy of each subsystem, $H(X,Y) \geq H(X), H(Y)$. In quantum information theory, this is not true. Actually, it can be shown that a bipartite system $S(Q,Q') = 0$ can still have $S(Q) = S(Q') > 0$. This is, for example, the case of the pure Bell state $|\rho_{\beta_0}\rangle\langle\rho_{\beta_0}|$. As we showed in Exercise 3.24, its entropy $S(\beta_0)$ – as for any other pure state – is zero. However, the reduced von Neumann entropies of the individual qubits (obtained by using the reduced density matrices (2.55) and (2.56)),

$$\rho_q = \frac{1}{2}\begin{bmatrix} 1 & 0 \\ 0 & 0 \end{bmatrix} + \frac{1}{2}\begin{bmatrix} 0 & 0 \\ 0 & 1 \end{bmatrix}, \qquad \rho_{q'} = \frac{1}{2}\begin{bmatrix} 1 & 0 \\ 0 & 0 \end{bmatrix} + \frac{1}{2}\begin{bmatrix} 0 & 0 \\ 0 & 1 \end{bmatrix}, \tag{3.78}$$

are greater than zero,

$$S(q) = \frac{1}{2} + \frac{1}{2} = 1\,, \qquad S(q') = \frac{1}{2} + \frac{1}{2} = 1\,. \tag{3.79}$$

Thus, as asserted, in quantum information theory,

$$S(\beta_0) < S(q), S(q')\,. \tag{3.80}$$

In classical information theory, we first defined conditional entropy by equation (3.28) and then you found that it satisfied property (3.32), $H(Y|X) = H(Y,X) - H(X)$. This last relation is used to define the *conditional von Neumann entropy* of a quantum bipartite system,

$$S(Q'|Q) = S(Q',Q) - S(Q)\,. \tag{3.81}$$

More explicitly,

$$S(\rho_{Q'}|\rho_Q) = -\mathrm{Tr}_{QQ'}\big[\rho_{QQ'}\log\rho_{QQ'}\big] - \mathrm{Tr}_Q\big[\mathrm{Tr}_{Q'}\rho_{QQ'}\log\mathrm{Tr}_{Q'}\rho_{QQ'}\big]\,. \tag{3.82}$$

Exercise 3.30. Prove that

$$S(Q'|Q) \leq S(Q')\,. \tag{3.83}$$

In analogy with the classical concept introduced in (3.38), we define the *mutual quantum information* by,

$$I(Q;Q') = S(Q) + S(Q') - S(Q,Q')\,. \tag{3.84}$$

Sometimes also denoted by $I(Q:Q')$. As for classical information, the mutual quantum information is semi-positive, $I(Q;Q') \geq 0$. It is equal to zero only when the two subsystems are uncorrelated, that is, when $\rho_{QQ'} = \rho_Q\,\rho_{Q'}$.

We can also define a relative quantum entropy that measures, as its classical counterpart, how different two probability distributions are. Given two probability distributions $Q = \{|Q_s\rangle, p_s\}$ and $\widetilde{Q} = \{|Q_{\tilde{s}}\rangle, p_{\tilde{s}}\}$, with respective density operators ρ_Q and $\rho_{\widetilde{Q}}$, their *relative von Neumann entropy* is

$$S(Q\|\widetilde{Q}) = S(\rho_Q\|\rho_{\widetilde{Q}}) = \mathrm{Tr}\big[\rho_Q(\log\rho_Q - \log\rho_{\widetilde{Q}})\big]\,. \tag{3.85}$$

Exercise 3.31. Show that, when written in diagonal form, the relative quantum entropy reduces to the classical formula (3.54).

Since $S(Q\|Q) = 0$, it follows that the larger $S(Q\|\widetilde{Q})$, the easier it is to distinguish between Q and $\widetilde{Q}$. Relative quantum entropy and mutual quantum information are related by

$$S(\rho_{QQ'}\|\rho_Q\rho_{Q'}) = I(Q;Q')\,. \tag{3.86}$$

Given two quantum channels $\Phi_Q(t,t_0)$ and $\Phi_{\widetilde{Q}}(t,t_0)$,

$$\rho_Q(t) = \Phi_Q(t,t_0)\rho_Q(t_0)\,, \qquad \rho_{\widetilde{Q}}(t) = \Phi_{\widetilde{Q}}(t,t_0)\rho_{\widetilde{Q}}(t_0)\,, \tag{3.87}$$

the relative quantum entropy always decreases after applying them,

$$S(\Phi_Q(t,t_0)\rho_Q(t_0)\|\Phi_{\widetilde{Q}}(t,t_0)\rho_{\widetilde{Q}}(t_0)) \leq S(\rho_Q(t_0)\|\rho_{\widetilde{Q}}(t_0))\,. \tag{3.88}$$

So, it is increasingly harder to distinguish between the two systems. For example, if $\rho_{QQ'} \mapsto \rho_Q$ and $\rho_{\widetilde{Q}\widetilde{Q}'} \mapsto \rho_{\widetilde{Q}}$, then

$$S(\rho_Q\|\rho_{\widetilde{Q}}) \leq S(\rho_{QQ'}\|\rho_{\widetilde{Q}\widetilde{Q}'})\,. \tag{3.89}$$

4 Algorithms and Secure Communication

In this section, we introduce the so called *Quantum Phase Estimation (QPE) algorithm* and the *Variational Quantum Eigensolver (VQE) algorithm*. Our approach, in both cases, is practical rather than purely theoretical.

Suppose a complex quantum system, for example, a molecule. Our knowledge of the ground-state energy of the system is crucial to understand how it interacts with external factors. This is something theorists know very well. In fact, for decades, physicists and chemists have been studying the electronic structure of molecules and have tried to find the ground-state energy of useful, but relatively simple, molecules. This endeavor has been fostered by the advent of powerful classical computers. However, despite considerable progress in this direction, traditional quantum chemistry and classical computers are inefficient to solve the ground-state energy problem of complex molecules which are of practical impact (such as the ones used in the pharmaceutical and food industries). By these means, it would take too long (exponential time) to have a complete knowledge of the ground-state energy of such molecules, needless to say an understanding of their chemical reactions. The two quantum algorithms we present here are attempts to solve this problem in practicable time (polynomial time).

It is worth mentioning that the fundamental difference between the two algorithms is practical. Actually, compared to the VQE algorithm, the QPE algorithm provides a more accurate estimate of the ground-state energy. However, while the QPE algorithm requires a fully-built quantum computer to operate, not accessible in the foreseeable future, the VQE algorithm integrates a classical part and a quantum subroutine that can be run in quantum computers deemed to be attainable in the near future.

Considerations concerning the future implementation of practical quantum algorithms have to do with the so-called *Noisy Intermediate-Scale Quantum (NISQ)* era we live in. In fact, despite the many advances in the technological front, due to the limited number and quality of the current qubits, the noisy gates and the undesirable interactions with the environment, we are still very far from achieving fully-quantum computers able to realize purely quantum algorithms. This is why

scientists are trying to design and develop algorithms with small quantum size and assign to powerful classical computers the rest of the tasks we already know they are good at. The VQE algorithm is an example of these more realistic algorithms scientists are trying to put in place.

We conclude this section with another useful application of quantum physics: secure communication by transferring information through quantum channels.

4.1 Quantum Phase Estimation

Suppose you want to describe, as precisely as possible, a highly complex time-independent quantum system. For example, a large molecule with many electrons. In principle, your goal is to find an exact analytic solution of the *time-independent Schrödinger equation*,

$$\hat{H}|\psi\rangle = E|\psi\rangle\,. \tag{4.1}$$

That is, given the Hamiltonian $\hat{H}$, you have to find the states $|\psi\rangle$ and the corresponding energies satisfying this equation. This problem, though, is in general very difficult and classical computational methods only provide approximate solutions.

Exercise 4.1. Do you remember how the equation (4.1) is obtained from the general time-dependent Schrödinger equation?

If you want to approach this problem using a quantum computer, the first thing you have to do is to model all the elements in equation (4.1) in terms of qubits and quantum circuits. Thus, we assume that you have found a qubit analog of the quantum system, that is, that you have a qubit Hilbert space $\mathcal{H}_Q$ containing all the state vectors $|Q\rangle$ that simulate any state $|\psi\rangle$ of the physical system. In addition, we assume that you have found a quantum circuit $\hat{H}_Q$ that models the Hamiltonian $\hat{H}$ of the system. The time-independent Schrödinger equation (4.1) then becomes an eingenvalue equation in the qubit space,

$$\hat{H}_Q|Q\rangle = E|Q\rangle\,, \tag{4.2}$$

where $|Q\rangle \in \mathcal{H}_Q \cong \mathbb{C}^{2^n}$ and $\hat{H}_Q\colon \mathcal{H}_Q \to \mathcal{H}_Q$. If we assume that the Hamiltonian is non-degenerate, then, there are 2^n eigenvalues E_ν with their corresponding eigenvectors $|E_\nu\rangle$,

$$\hat{H}_Q|E_\nu\rangle = E_\nu|E_\nu\rangle\,, \tag{4.3}$$

with $\nu = 0, \ldots, 2^n - 1$. The eigenbasis $\{|E_\nu\rangle\}$ can then be used to expand any state vector $|Q\rangle \in \mathcal{H}_Q$,

$$|Q\rangle = \sum_{\nu=0}^{2^n-1} c_\nu |E_\nu\rangle\,. \tag{4.4}$$

Exercise 4.2. Show that all the eigenvalues E_ν of $\hat{H}_Q$ are real and the set of eigenvectors $\{|E_\nu\rangle\}$ form an orthonormal basis of $\mathcal{H}_Q$.

Until here everything has been ideally simple. In reality, though, it is believed that the problem of finding the energy levels of a Hamiltonian is an exponentially hard problem, both for quantum and, all the more, for classical computers. Remember that a computational problem is said to be "easy" if the number of steps needed to solve it scales polynomially with the size of the input and it is said to be "hard" if it

scales exponentially. The QPE algorithm that we now present provides an estimate of the energy eigenvalues of a complex Hamiltonian.

Suppose we have been able to build the unitary transformation

$$U = e^{i\hat{H}_Q} \,. \tag{4.5}$$

It transforms an eigenvector of the Hamiltonian as follows,

$$U|E_\nu\rangle = e^{i\hat{H}_Q}|E_\nu\rangle = e^{iE_\nu}|E_\nu\rangle \,. \tag{4.6}$$

For convenience, we write $E_\nu = 2\pi\theta_\nu$, where $\theta_\nu \in [0,1]$ is called the *phase angle*,

$$U|E_\nu\rangle = e^{2\pi i\theta_\nu}|E_\nu\rangle \,. \tag{4.7}$$

(Note that to each eigenvector $|E_\nu\rangle$ there corresponds an angle θ_ν.) Our goal is to estimate the phase angles and, therefore, determine the eigenvalues E_ν of the Hamiltonian.

Exercise 4.3. Show that the eigenvalues of unitary transformations are complex numbers of unit magnitude.

Consider the following experiment,

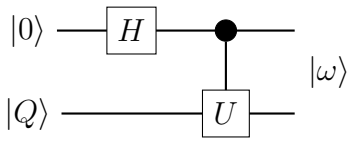

Fig. 3

The outgoing state $|\omega\rangle$ is obtained as follows,

$$\begin{aligned}
|0\rangle|Q\rangle &\longmapsto \frac{1}{\sqrt{2}}\sum_{k=1}^{2}|k\rangle\sum_{\nu=0}^{2^n-1}c_\nu|E_\nu\rangle \\
&\longmapsto \frac{1}{\sqrt{2}}\sum_k\sum_\nu c_\nu|k\rangle U^k|E_\nu\rangle \\
&= \frac{1}{\sqrt{2}}\sum_k\sum_\nu c_\nu|k\rangle\, e^{2\pi ik\theta_\nu}|E_\nu\rangle \,.
\end{aligned} \tag{4.8}$$

That is,

$$|\omega\rangle = \sum_\nu c_\nu\Big(\frac{1}{\sqrt{2}}\sum_k e^{2\pi ik\theta_\nu}|k\rangle\Big)|E_\nu\rangle \,. \tag{4.9}$$

At the end, you measure the upper qubit in the computational basis $\{|0\rangle\,,|1\rangle\}$. The

probability of obtaining the state $|j\rangle$ is,

$$\begin{aligned} P\big(\,|j\cdot\rangle\,\big) = |\langle j\ \cdot\ |\omega\rangle|^2 &= \sum_\mu \big|\langle j\, E_\mu\, |\omega\rangle\big|^2 \\ &= \sum_\mu \Big|\sum_\nu c_\nu \Big(\frac{1}{\sqrt{2}}\sum_k e^{2\pi i k\theta_\nu}\langle j|k\rangle\Big)\langle E_\mu|E_\nu\rangle\Big|^2 \\ &= \sum_\mu \Big|\sum_\nu c_\nu \Big(\frac{1}{\sqrt{2}}\sum_k e^{2\pi i k\theta_\nu}\delta_{jk}\Big)\delta_{\mu\nu}\Big|^2 \\ &= \sum_\mu \Big|c_\mu\Big(\frac{1}{\sqrt{2}}\,e^{2\pi i j\theta_\mu}\Big)\Big|^2 . \end{aligned} \tag{4.10}$$

We can simplify this result by assuming that the incoming state $|Q\rangle$ was prepared in an eigenstate $|E_\nu\rangle$, giving

$$\begin{aligned} P\big(\,|j\cdot\rangle\,\big) = |\langle j\ \cdot\ |\omega\rangle|^2 &= \sum_\mu \big|\langle j\, E_\mu\, |\omega\rangle\big|^2 \\ &= \sum_\mu \Big|\Big(\frac{1}{\sqrt{2}}\sum_k e^{2\pi i k\theta_\nu}\langle j|k\rangle\Big)\langle E_\mu|E_\nu\rangle\Big|^2 \\ &= \Big|\frac{1}{\sqrt{2}}\,e^{2\pi i j\theta_\nu}\Big|^2 \\ &= \frac{1}{2}\,. \end{aligned} \tag{4.11}$$

As expected, there is an equal probability of detecting the upper single qubit in the states $|0\rangle$ and $|1\rangle$. Instead of measuring the upper qubit, suppose we let it pass through an inverse quantum Fourier transform, $\mathrm{QFT}_1^\dagger$.

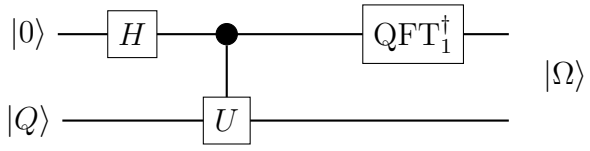

Fig. 4

In Box 4.2 of QC1 we saw that $\mathrm{QFT}_1 = H$, so $\mathrm{QFT}_1^\dagger = H^\dagger = H$. The outgoing state in this case is then,

$$\begin{aligned} |\omega\rangle \longmapsto |\Omega\rangle &= \sum_\nu c_\nu \Big(\frac{1}{\sqrt{2}}\sum_k e^{2\pi i k\theta_\nu}\,\mathrm{QFT}_1^\dagger|k\rangle\Big)|E_\nu\rangle \\ &= \sum_\nu c_\nu \Big(\frac{1}{\sqrt{2}}\sum_k e^{2\pi i k\theta_\nu} H|k\rangle\Big)|E_\nu\rangle\,. \end{aligned} \tag{4.12}$$

If we use the fact that,

$$H|k\rangle = \frac{1}{\sqrt{2}}\big(\,|0\rangle + (-1)^k\,|1\rangle\,\big) = \frac{1}{\sqrt{2}}\big(e^{-\pi ik0}\,|0\rangle + e^{-\pi ik1}\,|1\rangle\,\big)$$
$$= \frac{1}{\sqrt{2}}\sum_j e^{-\pi ikj}\,|j\rangle\ , \tag{4.13}$$

we finally get

$$|\Omega\rangle = \sum_\nu c_\nu\Big(\frac{1}{\sqrt{2}}\sum_k\ e^{2\pi ik\theta_\nu}\,\frac{1}{\sqrt{2}}\sum_j e^{-\pi ikj}\,|j\rangle\,\Big)|E_\nu\rangle$$
$$= \sum_\nu c_\nu\Big(\frac{1}{2}\sum_{k,j}\ e^{-\pi ik(j-2\theta_\nu)}|j\rangle\Big)|E_\nu\rangle\,. \tag{4.14}$$

We conclude the experiment by measuring the upper qubit in the computational basis. The probabilities are,

$$P\big(\,|\,l\,\cdot\,\rangle\,\big) = |\langle l\ \cdot\ |\Omega\rangle|^2 = \sum_\mu \big|\langle l\,E_\mu\,|\Omega\rangle\big|^2$$
$$= \sum_\mu \Big|\sum_\nu c_\nu\Big(\frac{1}{2}\sum_{k,j}\ e^{-\pi ik(j-2\theta_\nu)}\langle l|j\rangle\Big)\langle E_\mu|E_\nu\rangle\Big|^2$$
$$= \frac{1}{2^2}\sum_\mu \Big|c_\mu\sum_k\ e^{-\pi ik(l-2\theta_\mu)}\Big|^2\,. \tag{4.15}$$

In case the incoming state $|Q\rangle$ is in the eigenstate $|E_\nu\rangle$,

$$P\big(\,|\,l\,\cdot\,\rangle\,\big) = \frac{1}{2^2}\,\Big|\sum_{k=0}^{1}\ e^{-\pi ik(l-2\theta_\nu)}\Big|^2$$
$$= \frac{1}{2^2}\,\Big|\,1 + e^{-\pi i(l-2\theta_\nu)}\Big|^2\,. \tag{4.16}$$

Note that with the experimental result $P\big(\,|\,l\,\cdot\,\rangle\,\big)$, we can determine the angle θ_ν and, therefore, the energy eigenvalue.

Exercise 4.4. Find θ_ν in terms of $P\big(\,|\,l\,\cdot\,\rangle\,\big)$.

Let us extend the previous experiment to more than one unitary,

$$U_1|E_\nu\rangle = e^{2\pi i\theta_{1,\nu}}|E_\nu\rangle\,,\qquad U_2|E_\nu\rangle = e^{2\pi i\theta_{2,\nu}}|E_\nu\rangle\,, \tag{4.17}$$

where $0 \le \theta_{1,\nu}, \theta_{2,\nu} \le 1$. The experimental set-up is shown below,

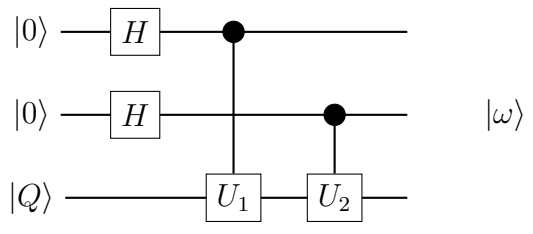

Fig. 5

The outgoing state $|\omega\rangle$ is obtained as follows,

$$
\begin{aligned}
|0\rangle|0\rangle|Q\rangle \longmapsto\longrightarrow & \frac{1}{\sqrt{2}}\sum_{i_1}|i_1\rangle\frac{1}{\sqrt{2}}\sum_{i_2}|i_2\rangle\sum_{\nu}c_\nu|E_\nu\rangle \\
&= \frac{1}{\sqrt{2^2}}\sum_{i_1,i_2}\sum_{\nu}c_\nu|i_1\ i_2\rangle|E_\nu\rangle \\
&\longmapsto\longrightarrow \frac{1}{\sqrt{2^2}}\sum_{i_1,i_2}\sum_{\nu}c_\nu|i_1\ i_2\rangle\, e^{2\pi i\, i_1\theta_{1,\nu}}|E_\nu\rangle \\
&\longmapsto\longrightarrow \frac{1}{\sqrt{2^2}}\sum_{i_1,i_2}\sum_{\nu}c_\nu|i_1\ i_2\rangle\, e^{2\pi i(i_1\theta_{1,\nu}+i_2\theta_{2,\nu})}|E_\nu\rangle\,. \qquad (4.18)
\end{aligned}
$$

That is,

$$
|\omega\rangle = \sum_{\nu}c_\nu\Big(\frac{1}{\sqrt{2^2}}\sum_{i_1,i_2}e^{2\pi i(i_1\theta_{1,\nu}+i_2\theta_{2,\nu})}|i_1\ i_2\rangle\Big)|E_\nu\rangle\,. \qquad (4.19)
$$

We now let the two-qubit system in the upper register enter an inverse quantum Fourier transform, $\mathrm{QFT}_2^\dagger$.

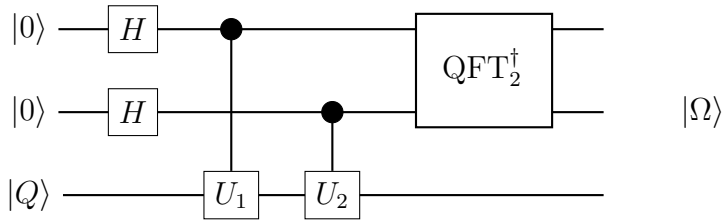

Fig. 6

The action of this gate was given in Box 4.3 of QC1,

$$
\mathrm{QFT}_2^\dagger|i_1\ i_2\rangle = \frac{1}{2}\sum_{y=0}^{3}e^{-\frac{\pi i}{2}(2i_1+i_2)}|y\rangle\,. \qquad (4.20)
$$

The state that exists the inverse QFT gate is then,

$$
\begin{aligned}
|\omega\rangle \longmapsto |\Omega\rangle &= \sum_{\nu}c_\nu\Big(\frac{1}{\sqrt{2^2}}\sum_{i_1,i_2}e^{2\pi i(i_1\theta_{1,\nu}+i_2\theta_{2,\nu})}\,\mathrm{QFT}_2^\dagger|i_1\ i_2\rangle\Big)|E_\nu\rangle \\
&= \sum_{\nu}c_\nu\Big(\frac{1}{\sqrt{2^2}}\sum_{i_1,i_2}e^{2\pi i(i_1\theta_{1,\nu}+i_2\theta_{2,\nu})}\,\frac{1}{2}\sum_{y=0}^{3}e^{-\frac{\pi i}{2}(2i_1+i_2)}|y\rangle\Big)|E_\nu\rangle \\
&= \sum_{\nu}c_\nu\Big(\frac{1}{2^2}\sum_{i_1,i_2}\sum_{y}e^{2\pi i(i_1\theta_{1,\nu}+i_2\theta_{2,\nu})}e^{-\frac{\pi i}{2}(2i_1+i_2)}|y\rangle\Big)|E_\nu\rangle\,. \qquad (4.21)
\end{aligned}
$$

This expression simplifies notably if we consider the special case $\theta_{1,\nu} = 2\theta_{2,\nu}$, that is, if $U_2^2 = U_1$. For simplicity, we define $\theta_{2,\nu} = \theta_\nu$. From here, $\theta_{1,\nu} = 2\theta_\nu$.

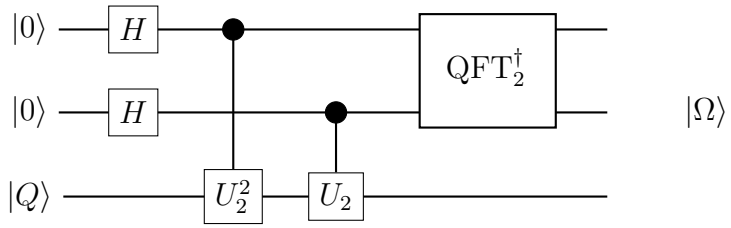

Fig. 7

The output state in this case is,

$$\begin{aligned}
|\Omega\rangle &= \sum_\nu c_\nu \Big(\frac{1}{2^2} \sum_{i_1,i_2} \sum_y e^{2\pi i (i_1 2\theta_\nu + i_2 \theta_\nu)} e^{-\frac{\pi i}{2}(2i_1+i_2)} |y\rangle \Big) |E_\nu\rangle \\
&= \sum_\nu c_\nu \Big(\frac{1}{2^2} \sum_{i_1,i_2} \sum_y e^{2\pi i (2i_1+i_2)\theta_\nu} e^{-\frac{\pi i}{2}(2i_1+i_2)} |y\rangle \Big) |E_\nu\rangle \\
&= \sum_\nu c_\nu \Big(\frac{1}{2^2} \sum_{x} \sum_y e^{2\pi i x \theta_\nu} e^{-\frac{\pi i}{2} x} |y\rangle \Big) |E_\nu\rangle \\
&= \sum_\nu c_\nu \Big(\frac{1}{2^2} \sum_{x,y} e^{-\frac{2\pi i}{4} x (y - 4\theta_\nu)} |y\rangle \Big) |E_\nu\rangle \,. \qquad (4.22)
\end{aligned}$$

In the third step, we have used the decimal expression of the binary string $i_1\, i_2$, namely, $x = 2i_1 + i_2$. We conclude the experiment by measuring the upper register. To compute the probabilities, though, we first note that since we have written the upper two-qubit system in decimal form, we have to do the same with the observational states. Denoting by $|z\,\cdot\rangle$ the observational states of the upper register, where $z = 0, 1, 2, 3$, the probabilities of observing them are,

$$\begin{aligned}
P\big(|\, z\cdot\rangle\big) = |\langle z\ \cdot\ |\Omega\rangle|^2 &= \sum_\mu \big| \langle z\, E_\mu\, |\Omega\rangle \big|^2 \\
&= \sum_\mu \Big| \sum_\nu c_\nu \Big(\frac{1}{2^2} \sum_{x,y} e^{-\frac{2\pi i}{4} x (y - 4\theta_\nu)} \langle z|y\rangle \Big) \langle E_\mu | E_\nu \rangle \Big|^2 \\
&= \frac{1}{(2^2)^2} \sum_\mu \Big| c_\mu \sum_x e^{-\frac{2\pi i}{2^2} x (z - 2^2 \theta_\mu)} \Big|^2 . \qquad (4.23)
\end{aligned}$$

If the input state $|Q\rangle$ is prepared in the eigenstate $|E_\nu\rangle$, the corresponding probabilities become

$$P\big(|\, z\cdot\rangle\big) = \frac{1}{(2^2)^2} \Big| \sum_{x=0}^{3} e^{-\frac{2\pi i}{2^2} x (z - 2^2 \theta_\nu)} \Big|^2 . \qquad (4.24)$$

The angles θ_ν, and consequently the eigenvalues E_ν of the Hamiltonian, can be determined using these probabilities.

Let us now generalize the above experiments to an arbitrary number A of ancillary qubits as shown in the figure below,

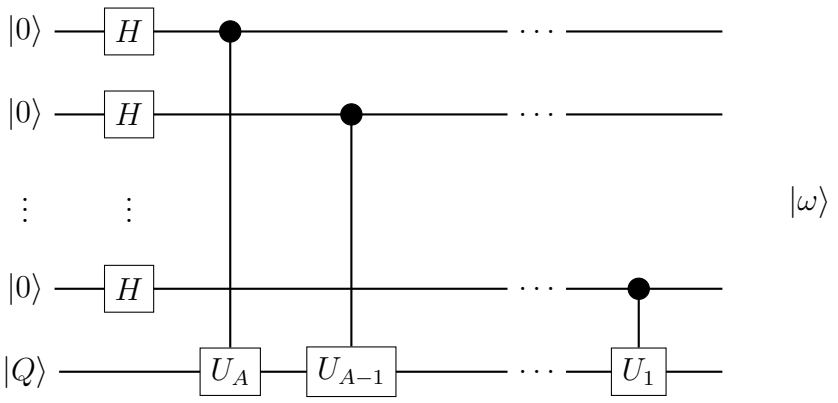

Fig. 8

The control gates are such that,

$$U_a|E_\nu\rangle = e^{2\pi i\theta_{a,\nu}}|E_\nu\rangle\,, \tag{4.25}$$

where $a = 1, 2, \ldots, A$. The state appearing on the right side of the circuit is,

$$\begin{aligned}|0\rangle^{\otimes A}|Q\rangle &\longmapsto \frac{1}{\sqrt{2^A}}\sum_{i_1,i_2,\ldots,i_A}|i_1\ i_2\ldots i_A\rangle\sum_\nu c_\nu|E_\nu\rangle \\ &\longmapsto \frac{1}{\sqrt{2^A}}\sum_{i_1,i_2,\ldots,i_A}\sum_\nu c_\nu|i_1\ i_2\ldots i_A\rangle\, e^{2\pi i(i_1\theta_{1,\nu}+i_2\theta_{2,\nu}+\ldots+i_A\theta_{A,\nu})}|E_\nu\rangle\,.\end{aligned} \tag{4.26}$$

In the previous examples, we saw that this expression simplifies considerably if we choose,

$$\theta_{A,\nu} = 2^0\theta_\nu\,,\quad \theta_{A-1,\nu} = 2\theta_\nu\,,\quad \ldots\quad \theta_{2,\nu} = 2^{A-2}\theta_\nu\,,\quad \theta_{1,\nu} = 2^{A-1}\theta_\nu\,. \tag{4.27}$$

In general,

$$\theta_{A-a+1,\nu} = 2^{a-1}\theta_\nu\,. \tag{4.28}$$

With this simplification, the outgoing state becomes

$$\begin{aligned}|\omega\rangle &= \frac{1}{\sqrt{2^A}}\sum_{i_1,i_2,\ldots,i_A}\sum_\nu c_\nu|i_1\ i_2\ldots i_A\rangle\, e^{2\pi i(i_1 2^{A-1}\theta_\nu+i_2 2^{A-2}\theta_\nu+\ldots+i_A\theta_\nu)}|E_\nu\rangle \\ &= \frac{1}{\sqrt{2^A}}\sum_{i_1,i_2,\ldots,i_A}\sum_\nu c_\nu|i_1\ i_2\ldots i_A\rangle\, e^{2\pi i(2^{A-1}i_1+2^{A-1}i_2+\ldots+i_A)\theta_\nu}|E_\nu\rangle\,.\end{aligned} \tag{4.29}$$

Finally, using the decimal representation of the binary string $i_1\ i_2\ldots i_A$,

$$x = 2^{A-1}i_1 + 2^{A-2}i_2 + \ldots + 2^0 i_A = \sum_{a=1}^{A} 2^{A-a}i_a\,, \tag{4.30}$$

we get,

$$|\omega\rangle = \sum_\nu c_\nu\Big(\frac{1}{\sqrt{2^A}}\sum_{x=0}^{2^A-1} e^{2\pi i x\theta_\nu}|x\rangle\Big)|E_\nu\rangle\,. \tag{4.31}$$

We now act on the upper register with the inverse quantum Fourier transform $\mathrm{QFT}_A^\dagger$,

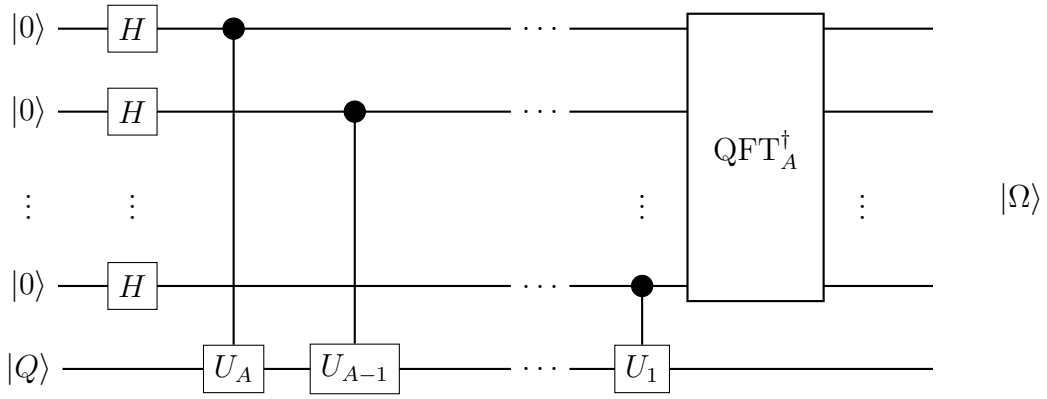

Fig. 9

According to equation (4.61) of QC1,

$$\mathrm{QFT}_{\mathrm{A}}^\dagger|x\rangle = \frac{1}{\sqrt{2^A}} \sum_{y=0}^{2^A-1} e^{-\frac{2\pi i}{2^A}xy}|y\rangle\,, \tag{4.32}$$

and thus,

$$\begin{aligned}
|\Omega\rangle &= \sum_\nu c_\nu \Big(\frac{1}{\sqrt{2^A}} \sum_x e^{2\pi i x\theta_\nu}\ \mathrm{QFT}_{\mathrm{A}}^\dagger|x\rangle \Big)|E_\nu\rangle \\
&= \sum_\nu c_\nu \Big(\frac{1}{\sqrt{2^A}} \sum_x e^{2\pi i x\theta_\nu} \Big(\frac{1}{\sqrt{2^A}} \sum_y e^{-\frac{2\pi i}{2^A}xy}|y\rangle \Big) \Big)|E_\nu\rangle \\
&= \sum_\nu c_\nu \Big(\frac{1}{2^A} \sum_{x,y} e^{-\frac{2\pi i}{2^A}x(y-2^A\theta_\nu)}|y\rangle \Big)|E_\nu\rangle\,.
\end{aligned} \tag{4.33}$$

Similar to the previous example where we had two ancillary qubits, the probability of measuring the state $|\,z\cdot\,\rangle$ in the upper register is given by

$$P\big(|\,z\cdot\,\rangle\big) = \frac{1}{(2^2)^A} \sum_\mu \Big| c_\mu \sum_x\ e^{-\frac{2\pi i}{2^A}x(z-2^A\theta_\mu)} \Big|^2\,. \tag{4.34}$$

In the special case that $|Q\rangle$ is in the eigenstate $|E_\nu\rangle$,

$$P\big(|\,z\cdot\,\rangle\big) = \frac{1}{(2^2)^A} \Big| \sum_{x=0}^{2^A-1} e^{-\frac{2\pi i}{2^A}x(z-2^A\theta_\nu)} \Big|^2\,. \tag{4.35}$$

In this last example we have assumed an arbitrary number A of ancillary qubits with no further considerations. Now, we want to discuss the implications that this has on the estimation of the phase angles θ_ν.

Consider the product $x\theta_\nu$ that appears in the exponential in (4.34). We know that $x = \sum_{a=1}^{A} 2^{A-a} i_a$ is given in terms of the number of ancillary qubits A. However, we have not considered the limitations that this finite number of ancillary qubits put

on the value of θ_ν. In fact, θ_ν can only have an approximate value, depending on the number of ancillary qubits A. The reason for this is that any fractional number between zero and one, such as θ_ν, can be written in the binary system as

$$\theta_\nu = \frac{b_1}{2^1} + \frac{b_2}{2^2} + \frac{b_3}{2^3} + \ldots + \frac{b_N}{2^N} + \ldots = \sum_{n=1}^{N} \frac{b_n}{2^n} + \ldots \,, \tag{4.36}$$

where $b_n = 0, 1$ and N is some positive integer number. In our case, with only A ancillary qubits,

$$\theta_\nu = \sum_{a=1}^{A} \frac{b_a}{2^a} + \ldots \,. \tag{4.37}$$

Thus, the greater the number of ancillary qubits, the better the approximation of θ_ν. Suppose that,

$$\theta_\nu^A = \sum_{a=1}^{A} \frac{b_a}{2^a} \,, \tag{4.38}$$

and $\theta_\nu = \theta_\nu^A + \delta_\nu$, where δ_ν indicates all the contributions that cannot be obtained using A ancillary qubits.

Using this, the outgoing state (4.33) becomes

$$|\Omega\rangle = \sum_\nu c_\nu \Big(\frac{1}{2^A} \sum_{x,y} e^{-\frac{2\pi i}{2^A} x(y - 2^A \theta_\nu^A)} e^{2\pi i x \delta_\nu} |y\rangle \Big) |E_\nu\rangle \,, \tag{4.39}$$

and the corresponding probability of measuring the state $|\, z \cdot \rangle$ in the upper register is,

$$P\big(|\, z \cdot \rangle\big) = \frac{1}{(2^2)^A} \sum_\mu \Big| c_\mu \sum_x e^{-\frac{2\pi i}{2^A} x(z - 2^A \theta_\mu^A)} e^{2\pi i x \delta_\mu} \Big|^2 \,. \tag{4.40}$$

Note that, if we measure the upper register and always get the same result, that is, if $P\big(|\, z \cdot \rangle\big) = 1$, we know with certainty that we have found the exact angle, $\theta_\mu^A = z/2^A$ (in other words, $\delta_\mu = 0$). In fact, in this case,

$$\begin{aligned} P\big(|\, z \cdot \rangle\big) &= \frac{1}{2^{2A}} \sum_\mu |c_\mu (1 + 1 + \ldots + 1)|^2 \\ &= \frac{1}{2^{2A}} \sum_\mu |c_\mu 2^A|^2 = \sum_\mu |c_\mu|^2 = 1 \,. \end{aligned} \tag{4.41}$$

On the contrary, the probability $P\big(|\, z \cdot \rangle\big) \neq 1$ when $\delta_\mu \neq 0$, indicating that we have not found the exact value of the phase angle.

To conclude, let us return to our original quantum problem. Suppose we start with an approximate ground-energy eigenstate provided by theoretical considerations,

$$|\tilde{E}_0\rangle = \sum_\nu c_\nu \, |E_\nu\rangle \,. \tag{4.42}$$

We let this state pass through the QPE circuit discussed above and measure the ancillary qubits. The probabilities are given in (4.40). The process is iterated until the lowest angle and, consequently, the lowest energy is found.

Even though the QPE algorithm shows that, in principle, a quantum computer can find the ground-state energy of a complex molecule in polynomial time, to run the algorithm requires a quantum computer that is not expected to be built in the near future. Because of this, new algorithms relying on future hybrid computers, part classic, part quantum, have been developed.

4.2 The Variational Quantum Eigensolver

Both, the Quantum Phase Estimation (QPE) algorithm, discussed in the previous subsection, and the Variational Quantum Eigensolver (VQE) algorithm, introduce here, can, in principle, solve the time-independent Schrödinger equation of a large quantum system. The fundamental difference between the two is that, while the former is an algorithm that requires machines built exclusively with quantum components, the latter operates with computers made of quantum as well classical components. The VQE is an example of what are called *classical-quantum hybrid algorithms*. These are the types of algorithms that are expected to first supersede classical algorithms given the current or near term available technology (NISQ era).

From the theoretical point of view, the difference between the two algorithms is that the VQE is a *variational algorithm* and the QPE is not. As we will see, this is connected to the future implementation of variational algorithms in NISQ devices mentioned above. Thus, the VQE uses the variational principle of quantum mechanics to find an approximate solution of the time-independent Schrödinger equation that describes a complex quantum system. Having stated the main differences between these two approaches, let us now explain how the VQE operates.

As usual, we denote by $|E_0\rangle$, $|E_1\rangle$, $|E_2\rangle$, ..., the set of orthonormal eigenstates of the Hamiltonian and assume that there is no degeneracy, with $E_0 < E_1 < E_2 < \ldots$. Any vector $|\psi\rangle$ corresponding to a physical state of the quantum system is thus written as a linear superposition of such eigenstates,

$$|\psi\rangle = \sum_{\nu} c_\nu |E_\nu\rangle \,. \tag{4.43}$$

The *variational theorem* affirms that the expectation value of the Hamiltonian for any physical state is always equal to or greater than the expectation value of the ground state,

$$\langle E_0|\hat{H}|E_0\rangle \leq \langle\psi|\hat{H}|\psi\rangle \,. \tag{4.44}$$

Only when $|\psi\rangle = |E_0\rangle$, that is, when $c_\nu = \delta_{\nu 0}$, do the equality holds.

Exercise 4.5. You should be able to prove this theorem.

Now, suppose we find a set of real parameters $\theta_1, \theta_2, \ldots, \theta_D$, that can be used to uniquely identify every state of the system,

$$|\psi\rangle = |\psi(\theta_1, \theta_2, \ldots, \theta_D)\rangle \,. \tag{4.45}$$

For instance, in QC1 we saw that there is a unique Bloch vector on the unit sphere, $|\psi\rangle = |\psi(\theta, \phi)\rangle$, where θ and ϕ are the spherical polar angles, associated to every state of a two-level system.

Exercise 4.6. Show that, in general, a state vector in a Hilbert space of dimension D requires $2(D-1)$ parameters.

To simplify the notation, let us introduce the parameters vector $\boldsymbol{\theta} = (\theta_1, \theta_2, \ldots, \theta_D)$ and the corresponding space of parameters $\mathcal{E}_\theta \ni \boldsymbol{\theta}$. We assume, moreover, that there is a $\boldsymbol{\theta}_0$ for which $|\psi(\boldsymbol{\theta}_0)\rangle = |E_0\rangle$. Thus, according to the variational theorem,

$$\langle\psi(\boldsymbol{\theta}_0)|\,\hat{H}\,|\psi(\boldsymbol{\theta}_0)\rangle \leq \langle\psi(\boldsymbol{\theta})|\,\hat{H}\,|\psi(\boldsymbol{\theta})\rangle\,. \tag{4.46}$$

When useful, we will use the following short-hand notation,

$$\langle\psi(\boldsymbol{\theta})|\,\hat{H}\,|\psi(\boldsymbol{\theta})\rangle = \langle\hat{H}\rangle(\boldsymbol{\theta}) = E(\boldsymbol{\theta})\,, \tag{4.47}$$

valid for any possible value of the parameter $\boldsymbol{\theta}$. The variational theorem, thus, affirms that,

$$\langle\hat{H}\rangle(\boldsymbol{\theta}_0) \leq \langle\hat{H}\rangle(\boldsymbol{\theta})\,, \tag{4.48}$$

or, equivalently,

$$E(\boldsymbol{\theta}_0) \leq E(\boldsymbol{\theta})\,. \tag{4.49}$$

Note that, since $E(\boldsymbol{\theta}_0)$ is nothing else that E_0, then

$$E_0 \leq E(\boldsymbol{\theta})\,. \tag{4.50}$$

In order to find a good approximate value of E_0, we start with a parameterized state $|\psi(\widetilde{\boldsymbol{\theta}})\rangle$, called the *trial* or *ansatz state*, and evaluate the expectation value $\langle\hat{H}\rangle(\widetilde{\boldsymbol{\theta}}) = E(\widetilde{\boldsymbol{\theta}})$. We do the same for other points in the parameters space $\mathcal{E}_\theta$ near $\widetilde{\boldsymbol{\theta}}$. That is, we compute $\langle\hat{H}\rangle(\Delta\widetilde{\boldsymbol{\theta}}) = E(\Delta\widetilde{\boldsymbol{\theta}})$. Now, since the expectation value of the Hamiltonian can be thought, mathematically speaking, as a multivariable real-valued function, $E\colon \mathcal{E}_\theta \to \mathbb{R}$, $\boldsymbol{\theta} \mapsto E(\boldsymbol{\theta}) \in \mathbb{R}$, we can use an optimization device to compare these results and provide us with the value $\boldsymbol{\theta}' \in \Delta\widetilde{\boldsymbol{\theta}}$ that corresponds to the minimum value of $E(\boldsymbol{\theta})$ for all $\boldsymbol{\theta} \in \Delta\widetilde{\boldsymbol{\theta}}$. We then use this vector and the space of parameters near it, $\Delta\widetilde{\boldsymbol{\theta}}'$, to compute $\langle\hat{H}\rangle(\Delta\widetilde{\boldsymbol{\theta}}') = E(\Delta\widetilde{\boldsymbol{\theta}}')$. We optimize again and iterate this process until we find a value $\langle\hat{H}\rangle(\boldsymbol{\theta}_*) = E(\boldsymbol{\theta}_*)$ that is as closed to E_0 as required by the nature of the problem (or, more precisely, until we meet the convergence criteria). Symbolically, we can write this process as follows,

$$\langle\hat{H}\rangle(\Delta\widetilde{\boldsymbol{\theta}}) \longmapsto \;\cdots\; \longmapsto \langle\hat{H}\rangle(\boldsymbol{\theta}_*) \gtrsim \langle\hat{H}\rangle(\boldsymbol{\theta}_0)\,, \tag{4.51}$$

that is,

$$E(\Delta\widetilde{\boldsymbol{\theta}}) \longmapsto \;\cdots\; \longmapsto E(\boldsymbol{\theta}_*) \gtrsim E(\boldsymbol{\theta}_0)\,. \tag{4.52}$$

In optimization theory, this iterative process is denoted by several equivalent notations,

$$\min_{\boldsymbol{\theta}} \langle\psi(\boldsymbol{\theta})|\hat{H}|\psi(\boldsymbol{\theta})\rangle = \min_{\boldsymbol{\theta}} \langle\hat{H}\rangle(\boldsymbol{\theta}) = \min_{\boldsymbol{\theta}}\, E(\boldsymbol{\theta}) = E_* \gtrsim E_0\,. \tag{4.53}$$

Now that we have properly posed the problem, let us see how the variational quantum eigensolver tackles it.

Exercise 4.7. Suppose a quantum system with Hamiltonian given by $\hat{H} = Z$. Starting with a generic parameterized state vector $|\psi(\tilde{\theta}, \tilde{\phi})\rangle$, explain in detail how the variational method proceeds to find the lowest-energy state.

As we said, the VQE is a hybrid quantum-classical algorithm. The quantum subroutine deals with the preparation of the quantum states and the evaluation of

the Hamiltonian expectation values. The optimization process is performed by a classical device (that we will not discuss here).

We assume that any state of the system, $|\psi(\boldsymbol{\theta})\rangle$, can be simulated by a multi-qubit state $|Q(\boldsymbol{\theta})\rangle$. In particular, the ansatz state $|\psi(\widetilde{\boldsymbol{\theta}})\rangle$ is modeled by $|Q(\widetilde{\boldsymbol{\theta}})\rangle$. The latter is prepared by letting pass n qubits in the state $|0\rangle^{\otimes n} = |\mathbf{0}\rangle$ through a parameterized circuit $U(\widetilde{\boldsymbol{\theta}})$,

$$U(\widetilde{\boldsymbol{\theta}})\,|\mathbf{0}\rangle = |Q(\widetilde{\boldsymbol{\theta}})\rangle\,. \tag{4.54}$$

This circuit $U(\widetilde{\boldsymbol{\theta}})$ is built by combining Pauli gates, single-qubit gate rotations, CNOT gates, and so on. After this, we need to evaluate the expectation value of the Hamiltonian. To do this, the Hamiltonian operator is modeled by a linear combination of Pauli operators. Remember that, in general, any Hamiltonian can be written as

$$\hat{H} = \sum h_{A_1\ldots A_n}\,\sigma_{A_1}\otimes\ldots\otimes\sigma_{A_n}\,, \tag{4.55}$$

where the σ_A's are the Pauli gates X, Y, Z, for $A = X, Y, Z$, or the identity operator, for $A = I$ (see QC1, equation (4.71)). That is, we measure

$$E(\widetilde{\boldsymbol{\theta}}) = \sum h_{A_1\ldots A_n}\langle\mathbf{0}|\,U^\dagger(\widetilde{\boldsymbol{\theta}})\,\sigma_{A_1}\otimes\ldots\otimes\sigma_{A_n}U(\widetilde{\boldsymbol{\theta}})\,|\mathbf{0}\rangle\,. \tag{4.56}$$

The VQE tells us that, from the computational point of view, it is more convenient to estimate each of the terms with a quantum device,

$$\langle\mathbf{0}|\,U^\dagger(\widetilde{\boldsymbol{\theta}})\,\sigma_{A_1}\otimes\ldots\otimes\sigma_{A_n}U(\widetilde{\boldsymbol{\theta}})\,|\mathbf{0}\rangle\,, \tag{4.57}$$

and let the sum be done by a classical computer. This process is then repeated for other parameters in the neighborhood of $\widetilde{\boldsymbol{\theta}}$,

$$\langle\mathbf{0}|\,U^\dagger(\Delta\widetilde{\boldsymbol{\theta}})\,\sigma_{A_1}\otimes\ldots\otimes\sigma_{A_n}U(\Delta\widetilde{\boldsymbol{\theta}})\,|\mathbf{0}\rangle\,, \tag{4.58}$$

All these results (obtained from the quantum device) are then sent to a classical optimizer. The optimizer tells us what is the set of parameters that minimizes this function. The process is iterated until we meet the convergence criteria. In summary,

$$\min_{\boldsymbol{\theta}}\sum h_{A_1\ldots A_n}\langle\mathbf{0}|\,U^\dagger(\boldsymbol{\theta})\,\sigma_{A_1}\otimes\ldots\otimes\sigma_{A_n}U(\boldsymbol{\theta})\,|\mathbf{0}\rangle = E_* \gtrsim E_0\,. \tag{4.59}$$

The above is an overview of the VQE that gives a broad idea of how the algorithm works. However, there are many aspects of the VQE that were not mentioned and are crucial to prove its future implementation and possible advantage. Let us mention just a few. Firstly, how do we choose the ansatz state? One way of doing it is by using conventional quantum chemistry theory. The theory will allow us to determine a state vector that is as close as possible to the correct ground state. However, the ansatz state cannot be too accurate either because, if not, the ansatz circuit to produce it could be too deep. Thus, a trade-off between accuracy and practicality is necessary. Secondly, what is the best way of representing the Hamiltonian? Above we have used the Pauli operators representation, however, ladder operators employed in second quantization are also of common use. Thirdly, what about error correction? NISQ algorithms are intended, by definition, to be run in machines that are not fault-tolerant quantum computers. But, is it not possible that the countless

components and integrations between classical and quantum components could render the computation completely useless due to the high amount of errors produced? Fourthly, and finally, is it true that the VQE is more efficient than classical algorithms running in modern supercomputers? This has not yet been proved formally (this is the reason why hybrid algorithms are also known as *heuristic algorithms*). All these are current research topics.

4.3 Quantum Cryptography

In modern times, the need for secure communication are manifold. One way to communicate securely is by encrypting the original message. That is, the text is modified by the sender according to a set of specific operations and, after transferring the message through a public communication channel, at the other end the receiver applies the inverse operations (decodes the message). The set of transformations that only the sender and the receiver know is what is called a *private (cryptographic) key*. In addition to the secrecy of the private key, the key cannot be inferred or obtainable by any means. It can be shown that, for binary messages, it suffices for the sender and the receiver to share a randomly generated binary key.

The problem is then to provide the sender and the receiver with the binary key, and to none else. The easier solution is, of course, to provide both of them with the key before they separate. However, this is sometimes impractical or simply impossible to realize in most situations. The question is then: how to distribute securely a binary key to two parties separated in space?

The first key distribution protocol based on the principles of quantum mechanics was proposed by Charles Bennett and Gilles Brassard in 1984. Several years later, in 1991, Arthur Ekert suggested an alternative key distribution scheme that also uses quantum mechanics. These two protocols are part of what nowadays is known as *quantum key distribution (QKD)*. Formally speaking, QKD is a subfield of *quantum cryptography*, a broader research area that studies the encryption and communication of information in a secure manner.

4.3.1 BB84 Protocol

Suppose that two parties, here denoted A and B, agree on the following communication protocol,

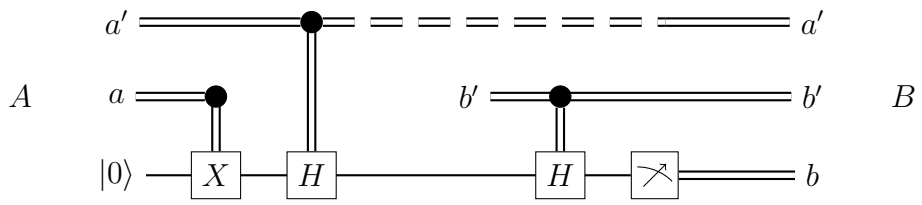

Fig. 10. BB84 protocol.

a and a' are binary bits obtained randomly and with equal probability from a trustful source. The double lines indicate that the Pauli X gate and the Hadamard gate are classical controlled-U gates determined by the values of a and a', respectively.

The quantum state that exits from the Hadamard gate of A is,

$$|0\rangle \xmapsto{\text{CC}X} |0 \oplus a\rangle = |a\rangle \xmapsto{\text{CC}H} H^{a'}|a\rangle\,. \tag{4.60}$$

Note that there are four possible states, depending on the values of a and a'.

Exercise 4.8. What are these four states?

Exercise 4.9. Show that

$$H^{a'}|a\rangle = \Big(\frac{1}{\sqrt{2}}\sum_j (-1)^{aj}\Big)^{a'}|a\rangle\,. \tag{4.61}$$

On the other side of the quantum communication channel, B receives the qubit $H^{a'}|a\rangle$ and applies to it a classical controlled-Hadamard gate determined by another random generated bit b',

$$H^{a'}|a\rangle \xmapsto{\text{CC}H} H^{b'}(H^{a'}|a\rangle) = H^{b'}H^{a'}|a\rangle\,. \tag{4.62}$$

Thus,

$$\begin{aligned} H^{b'}H^{a'}|a\rangle &= \Big(\frac{1}{\sqrt{2}}\sum_i (-1)^{ai}\Big)^{b'}\Big(\frac{1}{\sqrt{2}}\sum_j (-1)^{aj}\Big)^{a'}|a\rangle \\ &= \Big(\frac{1}{\sqrt{2}}\sum_i (-1)^{ai}\Big)^{b'+a'}|a\rangle = \Big(\frac{1}{\sqrt{2}}\sum_i (-1)^{ai}\Big)^{b'\oplus a'}|a\rangle\,. \end{aligned} \tag{4.63}$$

Exercise 4.10. Write explicitly all these states?

After this, B measures the qubit $H^{b'}H^{a'}|a\rangle$ in the computational basis $\{|0\rangle, |1\rangle\}$. The probability that B measures the state $|b\rangle$ is,

$$\begin{aligned} P_B\big(|b\rangle\big) &= \Big|\langle b|\Big(\frac{1}{\sqrt{2}}\sum_i (-1)^{ai}\Big)^{b'\oplus a'}|a\rangle\Big|^2 \\ &= \Big|\Big(\frac{1}{\sqrt{2}}\sum_i (-1)^{ai}\Big)^{b'\oplus a'}\langle b|a\rangle\Big|^2 = \frac{1}{2}\Big|\Big(\sum_i (-1)^{bi}\Big)^{b'\oplus a'}\Big|^2\,. \end{aligned} \tag{4.64}$$

B knows, even before any communication with A, that there are two possibilities: whether $a' = b'$ or $a' \neq b'$. If $a' = b'$, the state that exits from the Hadamard gate of B is,

$$H^{a'}H^{a'}|a\rangle = |a\rangle\,, \tag{4.65}$$

and the probability of measuring $|a\rangle$ is equal to

$$P_B\big(|b\rangle\big) = \big|\langle b|a\rangle\big|^2 = \big|\delta_{ab}\big|^2 = \delta_{ab}\,. \tag{4.66}$$

This result is rather obvious because $H^2 = I$ and, as stated by the protocol, the state $|a\rangle$ has equal probability of being $|0\rangle$ and $|1\rangle$. In the event that $a' \neq b'$,

$$H^{b'}H^{a'}|a\rangle = H|a\rangle = |(-1)^a\rangle\,. \tag{4.67}$$

That is, the outgoing state is $|+\rangle$ or $|-\rangle$ depending on the value of a. The probability that B measures $|b\rangle$ is then,

$$P_B\big(|b\rangle\big) = \frac{1}{2}\,. \tag{4.68}$$

In conclusion, every time $a' = b'$, the qubit measured by B turns out to be in $|b\rangle = |a\rangle$. In contrast, when $a' \neq b'$, B can measure $|0\rangle$ or $|1\rangle$ with equal probability.

At this point, using a classical channel, A sends to B the value of the random variable a'. If B notices that $a' \neq b'$, then, the result is discarded because the correspondence between b and a is not unique. However, when $a' = b'$, the two parties agree that $a = b$ and this result is kept to generate the shared key.

In order to generate a shared key that is long enough to communicate a complex message, the above procedure is repeated as many times as necessary. Alternatively, non-entangled single qubits can be used,

$$|0\rangle^{\otimes N} \xmapsto{(\mathrm{CC}X)^{\otimes N}} \bigotimes_{k=1}^{N} |a_k\rangle \xmapsto{(\mathrm{CC}H)^{\otimes N}} \bigotimes_{k=1}^{N} H^{a'_k}|a_k\rangle\,. \tag{4.69}$$

And, on the other end of the communication channel,

$$\bigotimes_{k=1}^{N} H^{a'_k}|a_k\rangle \xmapsto{(\mathrm{CC}H)^{\otimes N}} \bigotimes_{k=1}^{N} H^{b'_k}(H^{a'_k}|a_k\rangle)\,. \tag{4.70}$$

Exercise 4.11. Complete the argument.

This communication protocol is secure because any third party E between A and B would inevitably disturb the state of the communication qubit. Therefore, any deviation from the probability distribution found above would indicate the presence of an eavesdropper. Finally, it can be proved that the greater the number N, the higher the probability of knowing if the message was intercepted by a third party.

5 Error Correction and Fault Tolerance

How to detect and correct single qubit bit-flip errors is something we already discussed in our previous notes. There, we used the standard state vector formalism of quantum mechanics to describe the effects of the environment on a single qubit. However, since the qubit is an open system, a more appropriate approach (as you now know) is the density operator formalism. This is the first thing we do in this section. After this, we provide an introduction to the mathematics of the stabilizer formalism as applied to quantum circuits. We then revisit the repetition code for single-qubit errors, including bit flip and phase flip errors, and conclude with a quick introduction to fault-tolerant quantum computing (just enough for you to know what it is about and why it is so important for the future of quantum computers).

5.1 Single-Qubit Quantum Channels

Let us consider an n qubit interacting with its environment (see QC1, Subsection 5.1). We denote by $\rho_Q(t)$ and $\rho_e(t)$ their respective density operators at time t. Suppose that initially the environment is in a pure state,

$$\rho_e(t_0) = |e_0\rangle\langle e_0|\,, \tag{5.1}$$

and the qubit is decoupled from the environment. That is, the density operator of the entire system is

$$\rho_{Qe}(t_0) = \rho_Q(t_0) \otimes |e_0\rangle\langle e_0| \,. \tag{5.2}$$

As we know, the state operator of the qubit at any moment in time is obtained by tracing out the environment,

$$\rho_Q(t) = \mathrm{Tr}_{e_t}\big(\rho_{Qe}(t)\big) = \mathrm{Tr}_{e_t}\big(U(t,t_0)\rho_Q(t_0) \otimes |e_0\rangle\langle e_0|U^\dagger(t,t_0)\big) \,. \tag{5.3}$$

The usual convention is to denote the basis vectors of the Hilbert space of the environment at time t by $|e_t\rangle = |k\rangle_e$, so we have that

$$\begin{aligned} \rho_Q(t) &= \sum_k {}_e\langle k|U(t,t_0)|e_0\rangle \rho_Q(t_0) \big({}_e\langle k|U(t,t_0)|e_0\rangle\big)^\dagger \\ &= \sum_k E_k \rho_Q(t_0) E_k^\dagger \,. \end{aligned} \tag{5.4}$$

In the present context, the Kraus operators (see definition in (2.76))

$$E_k = {}_e\langle k|U(t,t_0)|e_0\rangle \,, \tag{5.5}$$

are called *error operators*. Remember that they satisfy the completeness relation

$$\sum_k E_k^\dagger E_k = 1 \,. \tag{5.6}$$

A *quantum channel*, or *quantum map*, is a superoperator $\Phi(t,t_0)$ that takes

$$\rho_Q(t_0) \longmapsto \rho_Q(t) = \Phi(t,t_0)\rho_Q(t_0) = \sum_k E_k \rho_Q(t_0) E_k^\dagger \,. \tag{5.7}$$

In this section, we want to apply these general considerations to the specific case of a single qubit that interacts with its environment. For notation convenience, from now on we will write $\rho_q(t_0) = \rho_0$ and $\rho_q(t) = \rho_t$. Thus,

$$\rho_t = \sum_k E_k \rho_0 E_k^\dagger \,. \tag{5.8}$$

Imagine that every time a single qubit $|q_0\rangle$ enters a communication channel, at the other end we receive a qubit $|q_t\rangle = \sigma_a|q_0\rangle$, where σ_a is one of the three Pauli operators X, Y, Z. The state operator of the qubit we receive is then

$$\rho_t = |q_t\rangle\langle q_t| = \sigma_a|q_0\rangle\langle q_0|\sigma_a = \sigma_a\rho_0\sigma_a \,. \tag{5.9}$$

Looking at (5.8), we recognize the error operator $E_k = \sigma_a$. Now, suppose that instead of $|q_t\rangle = \sigma_a|q_0\rangle$, we receive $|q_t\rangle = c\sigma_a|q_0\rangle$, where c is a real constant. This gives the state operator

$$\rho_t = |q_t\rangle\langle q_t| = c\sigma_a|q_0\rangle\langle q_0|c\sigma_a = c\sigma_a\rho_0 c\sigma_a \,. \tag{5.10}$$

In this case, the error operator is $E_k = c\sigma_a$. The value of c is obtained in the following two equivalent ways,

$$1 = \langle q_t|q_t\rangle = c^2\langle q_0|\sigma_a^\dagger \,\dagger_a\, q_0\rangle = c^2 \tag{5.11}$$

or

$$1 = \sum_k E_k^\dagger E_k = c^2 \sigma^\dagger \sigma_a = c^2 \,. \tag{5.12}$$

It thus follows that $c = 1$. In fact, c^2 is the probability that the initial state is affected by σ_a. More generally,

$$|q_t\rangle = \sqrt{1-p}\, I|q_0\rangle + \sqrt{p}\, \sigma_a|q_0\rangle \,, \tag{5.13}$$

where p is the probability that σ_a acts on the initial state and $1-p$ the probability that it stays unperturbed. The corresponding density operator is then,

$$\begin{aligned} \rho_t = |q_t\rangle\langle q_t| &= \big(\sqrt{1-p}\, I|q_0\rangle + \sqrt{p}\, \sigma_a|q_0\rangle\big)\big(\sqrt{1-p}\, I\langle q_0| + \sqrt{p}\, \langle q_0\sigma_a|\big) \\ &= (1-p)|q_0\rangle\langle q_0| + p\sigma_a|q_0\rangle\langle q_0|\sigma_a \\ &\quad + \sqrt{1-p}\sqrt{p}\, |q_0\rangle\langle q_0|\sigma_a + \sqrt{p}\, \sqrt{1-p}\, \sigma_a|q_0\rangle\langle q_0| \,. \end{aligned} \tag{5.14}$$

Exercise 5.1. Show that the last two terms cancel each other.

This gives,

$$\begin{aligned} \rho_t &= (1-p)\rho_0 + p\, \sigma_a\rho_0\sigma_a \,, \\ &= \big(\sqrt{1-p}\, I\big)\rho_0\big(\sqrt{1-p}\, I\big) + \big(\sqrt{p}\, \sigma_a\big)\rho_0\big(\sqrt{p}\, \sigma_a\big) \,. \end{aligned} \tag{5.15}$$

We recognize two Kraus/error operators, $E_1 = \sqrt{1-p}\, I$ and $E_2 = \sqrt{p}\, \sigma_a$.

Exercise 5.2. Show that these error operators satisfy the completeness relation.

An example of this, is the *bit-flip channel* $\Phi_X(t, t_0) = \Phi_X$. It has $\sigma_a = X$,

$$\rho_t = \Phi_X \rho_0 = (1-p)\rho_0 + pX\rho_0 X \,. \tag{5.16}$$

The question we now want to answer is: how is the Bloch vector of the original qubit transformed under the action of the bit-flip channel? To this end, we use equation (2.34), $\rho = 1/2\,(I + \mathbf{B}\cdot\boldsymbol{\sigma})$, to write both the initial and final state operators ρ_0 and ρ_t and then compare the respective values of $\mathbf{B}_0$ and $\mathbf{B}_t$. For the bit-flip channel,

$$\begin{aligned} \rho_t &= \frac{1}{2}\big(I + \mathbf{B}_t\cdot\boldsymbol{\sigma}\big) \\ &= (1-p)\rho_0 + pX\rho_0 X \\ &= (1-p)\,\frac{1}{2}\big(I + \mathbf{B}_0\cdot\boldsymbol{\sigma}\big) + pX\,\frac{1}{2}\big(I + \mathbf{B}_0\cdot\boldsymbol{\sigma}\big)X \\ &= \frac{1}{2}\big(I + \mathbf{B}_0\cdot\boldsymbol{\sigma} - p\,\mathbf{B}_0\cdot\boldsymbol{\sigma} + p\,X\,\mathbf{B}_0\cdot\boldsymbol{\sigma}\,X\big) \,. \end{aligned} \tag{5.17}$$

Thus,

$$\mathbf{B}_t\cdot\boldsymbol{\sigma} = \mathbf{B}_0\cdot\boldsymbol{\sigma} - p\,\mathbf{B}_0\cdot\boldsymbol{\sigma} + p\,X\,\mathbf{B}_0\cdot\boldsymbol{\sigma}\,X \tag{5.18}$$

Exercise 5.3. Use this result to show that

$$\begin{aligned} \mathbf{B}_t\cdot\boldsymbol{\sigma} &= (B_t)_x X + (B_t)_y Y + (B_t)_z Z \\ &= (B_0)_x X + (1-2p)(B_0)_y Y + (1-2p)(B_0)_z Z \,. \end{aligned} \tag{5.19}$$

From here it follows that the Bloch vector is modified by the bit-flip channel according to,

$$\mathbf{B}_0 \longmapsto \mathbf{B}_t = \begin{bmatrix} (B_t)_x \\ (B_t)_y \\ (B_t)_z \end{bmatrix} = \begin{bmatrix} (B_0)_x \\ (1-2p)(B_0)_y \\ (1-2p)(B_0)_z \end{bmatrix} = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1-2p & 0 \\ 0 & 0 & 1-2p \end{bmatrix} \mathbf{B}_0 \,. \tag{5.20}$$

For example, for the initial Bloch vectors,

$$\begin{aligned} \mathbf{B}_0 = [\,1\;0\;0\,]^T &\longmapsto \mathbf{B}_t = [\,1\;0\;0\,]^T \,, \\ \mathbf{B}_0 = [\,0\;1\;0\,]^T &\longmapsto \mathbf{B}_t = [\,0\;(1-2p)\;0\,]^T \,, \\ \mathbf{B}_0 = [\,0\;0\;1\,]^T &\longmapsto \mathbf{B}_t = [\,0\;0\;(1-2p)\,]^T \,. \end{aligned} \tag{5.21}$$

Assuming that $p < 1/2$, then $0 < 1 - 2p < 1$. We note that the bit-flip channel contracts the y and z components of the Bloch vector associated to a single qubit (the length of the x component does not change).

Exercise 5.4. Work out the details for the *phase-flip channel* ($\sigma_a = Z$), as well as for the *bit-phase-flip channel* ($\sigma_a = Y$).

5.2 Stabilizers Circuits

Let $|q\rangle$ represent the state of a single qubit, $|q\rangle \in \mathcal{H}_q \cong \mathbb{C}^2$. As we saw in QC1, Box 4.3, any unitary transformation U on a single-qubit state vector $|q\rangle$ can be written as a linear combination of elements of the single-qubit Pauli group $\mathcal{P}_1 = \{\sigma_A; \pm i, \pm 1\}$, where $A = I, X, Y, Z$. That is,

$$U = \sum_A a_A \, \sigma_A \,. \tag{5.22}$$

In general, of course, a state vector $|q\rangle \in \mathcal{H}_q$ will change under the action of an operator U, $U|q\rangle \neq |q\rangle$. However, there are some specific vectors $|q\rangle_S \in \mathcal{H}_q$ that are left unchanged by some transformations U_S,

$$U_S |q\rangle_S = |q\rangle_S \,. \tag{5.23}$$

In algebraic jargon, we say that U_S leaves $|q\rangle_S$ "fixed" or that U_S "stabilizes" $|q\rangle_S$.

Exercise 5.5. What are the state vectors in $\mathcal{H}_q$ that are stabilized by X, Y and Z? Show that these vectors are unique. Conversely, show that X, Y and Z are, in fact, the only operators that stabilize these vectors.

Exercise 5.6. Provide a visual representation of the previous exercise in terms of the Bloch sphere.

Consider now a two qubit $|q_2\rangle \in \mathcal{H}_{q_2} \cong \mathbb{C}^4$. Any unitary transformation U on the two qubit will be written as a linear combination of elements of the 2-qubit Pauli group $\mathcal{P}_2 = \{\sigma_A; \pm i, \pm 1\}^{\otimes 2}$,

$$U = \sum_{A,B} a_{AB} \, \sigma_A \, \sigma_B \,. \tag{5.24}$$

In general,

$$U|q_2\rangle = \sum_{A,B} a_{AB}\, \sigma_A\, \sigma_B \sum_{i,j} \alpha_{ij}|i\rangle|j\rangle = \sum_{A,B}\sum_{i,j} a_{AB}\sigma_A|i\rangle\, \sigma_B|j\rangle\,. \tag{5.25}$$

Again, some two qubits will be modified by some unitaries, some will not. If

$$U_S|q_2\rangle_S = |q_2\rangle_S\,, \tag{5.26}$$

we say that U_S stabilizes $|q_2\rangle_S$. For example, the two qubit $|0\,0\rangle$ is stabilized by the set of operators $\{I\,I, Z\,I, I\,Z, Z\,Z\}$ and $|+\,+\rangle$ by $\{I\,I, X\,I, I\,X, X\,X\}$. Note that the sum of two or more operators that stabilize a qubit does not stabilize it. By convention, we will only consider operators that are products of Pauli matrices and the identity,

$$M_{AB} = \sigma_A\, \sigma_B\,, \tag{5.27}$$

for some A and B.

Exercise 5.7. Show that $|+\,0\,\rangle$ is stabilized by $\{I\,I, X\,I, I\,Z, X\,Z\}$.

Exercise 5.8. Show that the sets of operators that stabilize $|0\,0\rangle$, $|+\,+\rangle$ and $|+\,0\,\rangle$ are, each of them individually, subgroups of the Pauli group $\mathcal{P}_2$.

Before moving to higher qubits, let us find some operators that stabilize the Bell states $|\beta_{ij}\rangle \in \mathcal{H}_{q_2} \cong \mathbb{C}^4$, where $i, j = 0, 1$,

$$|\beta_{ij}\rangle = \frac{1}{\sqrt{2}}\big[|0\;i\rangle + (-1)^j|1\;\bar{i}\rangle\big]\,. \tag{5.28}$$

(See QC1, equation (4.64).) Since $X|i\rangle = |\bar{i}\rangle$,

$$\begin{aligned} X\,X|\beta_{ij}\rangle &= \frac{1}{\sqrt{2}}\big[|1\;\bar{i}\rangle + (-1)^j|0\;i\rangle\big] \\ &= \frac{(-1)^j}{\sqrt{2}}\big[(-1)^j|1\;\bar{i}\rangle + |0\;i\rangle\big] = (-1)^j|\beta_{ij}\rangle\,. \end{aligned} \tag{5.29}$$

Recalling that (with $i = \sqrt{-1}$),

$$\begin{aligned} Y|j\rangle &= (-1)^j i\,|\bar{j}\rangle\,, \\ Y|\bar{j}\rangle &= (-1)^{\bar{j}} i\,|\bar{\bar{j}}\rangle = (-1)^{j+1} i\,|j\rangle = -(-1)^j i\,|j\rangle\,, \end{aligned} \tag{5.30}$$

we get,

$$\begin{aligned} Y\,Y|\beta_{kj}\rangle &= \frac{1}{\sqrt{2}}\big[i|1\rangle(-1)^k i\,|\bar{k}\rangle + (-1)^j(-1)\,i\,|0\rangle(-1)(-1)^k i\,|k\rangle\big] \\ &= \frac{-(-1)^k}{\sqrt{2}}\big[|1\;\bar{k}\rangle + (-1)^j|0\;k\rangle\big] = -(-1)^{k+j}|\beta_{kj}\rangle\,. \end{aligned} \tag{5.31}$$

Finally, since $Z|i\rangle = (-1)^i|i\rangle$,

$$\begin{aligned} Z\,Z|\beta_{ij}\rangle &= \frac{1}{\sqrt{2}}\big[|0\rangle(-1)^i|i\rangle + (-1)^j(-1)|1\rangle(-1)^{\bar{i}}|\bar{i}\rangle\big] \\ &= \frac{(-1)^i}{\sqrt{2}}\big[|0\;i\rangle + (-1)^{j+1+\bar{i}-i}|1\;\bar{i}\rangle\big] = (-1)^i|\beta_{ij}\rangle\,. \end{aligned} \tag{5.32}$$

That is, a Bell state $|\beta_{ij}\rangle$ is stabilized by the following operators,

$$\big[(-1)^j X\, X\big]|\beta_{ij}\rangle = |\beta_{ij}\rangle\,, \tag{5.33}$$

$$\big[-(-1)^{i+j} Y\, Y\big]|\beta_{ij}\rangle = |\beta_{ij}\rangle\,, \tag{5.34}$$

$$\big[(-1)^i Z\, Z\big]|\beta_{ij}\rangle = |\beta_{ij}\rangle\,. \tag{5.35}$$

Exercise 5.9. Show that the operators $I\,I$, $X\,X$, $Y\,Y$ and $Z\,Z$ commute between them.

Exercise 5.10. Show that the operators in each of the sets of Exercise 5.8 commute between them.

For three qubits, the situation is similar. Any three qubit $|q_3\rangle$ is stabilized by a product of sigma matrices and the identity,

$$M_{ABC}|q_3\rangle = \sigma_A\,\sigma_B\,\sigma_C|q_3\rangle = |q_3\rangle\,, \tag{5.36}$$

for some A, B and C.

Exercise 5.11. Find the operators that stabilize the GHZ state $1/\sqrt{2}\,(|0\,0\,0\rangle + |1\,1\,1\rangle)$.

We generalize to n qubits as follows. For any n qubit $|q_n\rangle \in \mathcal{H}_{q_n}$, there are operators $M_{A_1\ldots A_n}$ in the n-qubit Pauli group $\mathcal{P}_n = \{\sigma_A; \pm i, \pm 1\}^{\otimes n}$ that leave the qubit unchanged,

$$M_{A_1\ldots A_n}|q_n\rangle = \sigma_{A_1}\ldots\sigma_{A_n}|q_n\rangle = |q_n\rangle\,, \tag{5.37}$$

for some $A_1, \ldots, A_n$. In other words, $|q_n\rangle$ is an eigenvector of $M_{A_1\ldots A_n}$ with eigenvalue one. The set of all these operators form a commutative subgroup of the n-qubit Pauli group known as the *stabilizer group* of the n qubit $|q_n\rangle$,

$$\{M_{A_1\ldots A_n}\} = \mathcal{S}(|q_n\rangle) \subset \mathcal{P}_n\,. \tag{5.38}$$

Each operator in the stabilizer group, $M_{A_1\ldots A_n} \in \mathcal{S}(|q_n\rangle)$, is called a *stabilizer operator* or simply a *stabilizer* .

Exercise 5.12. Justify why they form a subgroup of $\mathcal{P}_n$.

Exercise 5.13. Why is it that $(M_{A_1\ldots A_n})^2 = I_n$?

That every stabilizer group is commutative is also easy to show. Suppose that M_1 and M_2 are two stabilizers of the n qubit $|q_n\rangle$. We must then have

$$M_1 M_2|q_n\rangle = M_1(M_2|q_n\rangle) = M_1|q_n\rangle = |q_n\rangle\,, \tag{5.39}$$

$$M_2 M_1|q_n\rangle = M_2(M_1|q_n\rangle) = M_2|q_n\rangle = |q_n\rangle\,. \tag{5.40}$$

That is, for any two stabilizers M_1 and M_2,

$$M_1 M_2|q_n\rangle = M_2 M_1|q_n\rangle\,. \tag{5.41}$$

Assuming that $|q_n\rangle_S$ is not the null vector, we conclude that

$$[M_1, M_2] = 0\,. \tag{5.42}$$

If we consider several qubits in $\mathcal{H}_{q_n}$, each with its own stabilizer group, the common elements of these stabilizer groups form a *stabilizer code*. A stabilizer code is, thus, the set of operators in $\mathcal{P}_n$ that stabilize the subspace spanned by these qubits.

Exercise 5.14. What are the stabilizer codes of the subspaces spanned by the vectors given in Exercise 5.8?

Now that we know how to apply the stabilizer formalism to qubits, we want to show how it is used to describe the gates of a quantum circuit. Suppose an n qubit with stabilizer M,

$$M|q_n\rangle = |q_n\rangle\,. \tag{5.43}$$

If the qubit $|q_n\rangle$ enters a gate U, at the other end will exit the qubit

$$U|q_n\rangle = UM|q_n\rangle = UMU^\dagger(U|q_n\rangle)\,. \tag{5.44}$$

That is, when the incoming qubit $|q_n\rangle$ is stabilized by M, then, the outgoing state $U|q_n\rangle$ is stabilized by $UMU^\dagger$.

This indicates that instead of analyzing a quantum circuit by referring to the evolution of the input state as it moves through the circuit, we can equivalently describe it by the evolution of the stabilizer.

For example, instead of $|0\rangle \overset{H}{\longmapsto} |+\rangle$, we can use $Z \overset{H}{\longmapsto} X$,

Fig. 11

Similarly, we substitute $|0\rangle \overset{Y}{\longmapsto} i\,|1\rangle$ by $Z \overset{Y}{\longmapsto} -Z$,

Fig. 12

Exercise 5.15. Show that the state vector description $|+\rangle \overset{S}{\longmapsto} S|+\rangle$ is equivalent to $X \overset{S}{\longmapsto} Y$ in the stabilizer formalism.

Now, to produce entangled states, we need multi-qubit gates and they too have to be represented in the language of stabilizers.

Recall that a generic controlled-U gate transforms

$$|i\rangle|j\rangle \overset{CU}{\longmapsto} |i\rangle\, U^i|j\rangle\,, \tag{5.45}$$

where $|i\rangle$ is the control qubit and $|j\rangle$ the target qubit.

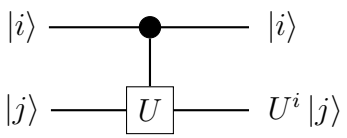

Fig. 13. The controlled-U gate.

Since the control qubit $|i\rangle$ can be $|0\rangle$ or $|1\rangle$, it will be stabilized by Z or $-Z$, respectively. Denoting these stabilizers by $(-1)^i Z$, and using a similar for the target

qubit, we have that the incoming state is stabilized by $(-1)^i Z \otimes (-1)^j Z$. On the other side of the CU gate, we have that the upper qubit remains unchanged so its stabilizer is still $(-1)^i Z$. The lower qubit, though, is transformed according to $U^i|j\rangle$. But, as we saw in (5.44), if the qubit $|j\rangle$ is stabilized by $(-1)^j Z$ and it is transformed by the unitary $U^i|j\rangle$, the new state will be stabilized by $U^i(-1)^j Z U^{i\dagger}$. The stabilizer representation of the CU gate is thus,

$$(-1)^i Z(-1)^j \otimes Z \xmapsto{CU} (-1)^i Z \otimes U^i(-1)^j Z U^{i\dagger} \,. \tag{5.46}$$

For example, for a CNOT gate, where $U = X$,

$$(-1)^i Z \otimes (-1)^j Z \xmapsto{\text{CNOT}} (-1)^i Z \otimes X^i(-1)^j Z X^i \,. \tag{5.47}$$

We can check this result by looking at the following equivalence between the state vector and stabilizer formalism,

$$\begin{aligned} |0\rangle|j\rangle &\xmapsto{\text{CNOT}} |0\rangle|j\rangle \,, \qquad Z \otimes (-1)^j Z \xmapsto{\text{CNOT}} Z \otimes X^i(-1)^j Z X^i \,, \\ |1\rangle|j\rangle &\xmapsto{\text{CNOT}} |1\rangle|j\rangle \,, \qquad -Z \otimes (-1)^j Z \xmapsto{\text{CNOT}} -Z \otimes X^i(-1)^j Z X^i \,. \end{aligned} \tag{5.48}$$

Exercise 5.16. Check this result by considering all possible cases.

As a final example, consider again the gate that creates the $|\beta_0\rangle$ state,

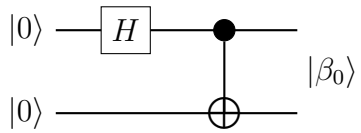

Fig. 14. Reproduction of Figure 1.

At every step of the process the initial state changes as follows,

$$\begin{aligned} |0\,0\rangle &\xmapsto{H\otimes I} \frac{1}{\sqrt{2}}\big(|0\,0\rangle + |1\,0\rangle\big) = |+0\rangle \\ &\xmapsto{\text{CNOT}} \frac{1}{\sqrt{2}}\big(|0\,0\rangle + |1\,1\rangle\big) = |\beta_0\rangle \,. \end{aligned} \tag{5.49}$$

In the stabilizer representation,

$$Z\,Z \xmapsto{H\otimes I} X\,Z \xmapsto{\text{CNOT}} X\,X \,. \tag{5.50}$$

Exercise 5.17. Repeat this for the general Bell state $|\beta_{ij}\rangle$.

If you remember, at the beginning of our discussion on stabilizers we said that the two qubit $|0\,0\rangle$ was stabilized by the set of operators $\{I\,I, Z\,I, I\,Z, Z\,Z\}$. But in (5.50) we only have $Z\,Z$! The natural question then is: where did the other stabilizers go? The same can be said about the states $|+0\rangle$ and $|\beta_0\rangle$, with stabilizer groups $\{I\,I, X\,I, I\,Z, X\,Z\}$ and $\{I\,I, X\,X, Z\,Z\}$, respectively. How do we take into account all the other group elements? One obvious way would be to write,

$$\{I\,I, Z\,I, I\,Z, Z\,Z,\} \xmapsto{H\otimes I} \{I\,I, X\,I, I\,Z, X\,Z\} \xmapsto{\text{CNOT}} \{I\,I, X\,X, Z\,Z\} \,. \tag{5.51}$$

However, we can use the fact that stabilizers form a commutative group to only indicate the *group generators*. For example, since

$$I\,I = (Z\,I)(Z\,I)\,, \qquad Z\,Z = (Z\,I)(I\,Z)\,, \tag{5.52}$$

we only need the operators $Z\,I$ and $I\,Z$ to generate the group $\{I\,I, Z\,I, I\,Z, Z\,Z,\}$. Similarly,

$$I\,I = (X\,I)(X\,I)\,, \qquad Z\,X = (X\,I)(I\,Z)\,, \tag{5.53}$$

and, thus, we pick the operators $X\,I$ and $I\,Z$ to generate the stabilizer group $\{I\,I, X\,I, I\,Z, X\,Z\}$. Finally, for $\{I\,I, X\,X, Z\,Z\}$ we pick $X\,X$ and $Z\,Z$ because

$$I\,I = X\,X = Z\,Z\,. \tag{5.54}$$

Thus, the process that creates the $|\beta_0\rangle$ state can be represented in the form

$$\{Z\,I, I\,Z\} \overset{H\otimes I}{\longmapsto} \{X\,I, I\,Z\} \overset{\text{CNOT}}{\longmapsto} \{X\,X, Z\,Z\}\,. \tag{5.55}$$

Exercise 5.18. Reproduce a similar argument for the CU gate discussed above (in particular, the CNOT gate).

5.3 Stabilizer QEC Codes

Having established how the stabilizer formalism describes quantum circuits, our next task is to explain how it can be used to detect and correct errors that can occur during the transmission of a qubit. We first examine the three-qubit repetition code for bit-flip errors discussed in QC1, Subsection 5.4.

The first step is to create identical copies of the single-qubit basis states $|0\rangle$ and $|1\rangle$ of $\mathcal{H}_q \cong \mathbb{C}^2$,

$$|i\rangle \longmapsto |i\rangle_L = |i\;i\;i\rangle\,. \tag{5.56}$$

The initial single qubit $|q\rangle$ in $\mathcal{H}_q \cong \mathbb{C}^2$ then becomes a three qubit $|q\rangle_L$ in an extended Hilbert space $\mathcal{H}_{q_3} \cong \mathbb{C}^8$ (the so called logical qubit Hilbert space),

$$|q\rangle = \sum_i \alpha_i|i\rangle \longmapsto \sum_i \alpha_i|i\rangle_L = \sum_i \alpha_i|i\;i\;i\rangle = |q\rangle_L\,. \tag{5.57}$$

The fact that a bit-flip error can occur to any of the three physical qubits is taken into account by considering the following state vector,

$$|q\rangle_L \longmapsto |q\rangle_E = \sum_i \alpha_i|i\;i\;i\rangle + \sum_i \alpha_i|\bar{i}\;i\;i\rangle + \sum_i \alpha_i|i\;\bar{i}\;i\rangle + \sum_i \alpha_i|i\;i\;\bar{i}\rangle\,. \tag{5.58}$$

With the help of two ancilla qubits, each physical qubit is passed through a CNOT gate to measure the error syndrome and correct the qubit that has been flipped.

The stabilizer formalism approaches the problem of correcting bit-flip errors differently. The error syndrome is measured by gates that act on the physical qubits, without the need of any ancilla qubit.

Consider a generic vector $|i\;j\;k\rangle$ of (5.58). Suppose we let it pass through the gate $Z\,Z\,I$,

$$Z\,Z\,I\,|i\;j\;k\rangle = (-1)^{i+j}|i\;j\;k\rangle\,. \tag{5.59}$$

We take note of the result. The eigenvalue of $Z\,Z\,I$ is 1 when $i = j$, in which case the outgoing state is $|i\ i\ k\rangle$, and is equal to -1 when $i \neq j$, in which case the outgoing state is $|i\ \bar{i}\ k\rangle$. We now take this qubit and send it to a $I\,Z\,Z$ gate. In general,

$$I\,Z\,Z\,|i\ j\ k\rangle = (-1)^{j+k}|i\ j\ k\rangle\,. \tag{5.60}$$

Again, the eigenvalue of the operator $I\,Z\,Z$ can be plus or minus one, depending of the values of j and k. It is 1 when $j = k$ and -1 when $j \neq k$. We then have four possibilities: $i = j$ followed by $j = k$ or $j \neq k$, and $i \neq j$ followed by $j = k$ or $j \neq k$. Only two of these possibilities correspond to errors that have to be corrected ($i \neq j$ followed by $j = k$ and $i = j$ followed by $j \neq k$). To illustrate how this works, consider the following scenario. If we measure 1 for $Z\,Z\,I$, then we must have $i = j$. Since we are considering only one bit flip at most, this means that there is no error in the first and second qubits. Suppose we then measure -1 for $I\,Z\,Z$. This means that $j \neq k$. But, in the first measurement we found $i = j$, thus $i = j \neq k$. We conclude that the third qubit has been flipped and we must correct it by applying a $I\,I\,X$ gate. Finally, note that, since $Z\,Z\,I$ and $I\,Z\,Z$ commute, we can first apply $Z\,Z\,I$ and then $I\,Z\,Z$, or vice versa. The final result is the same.

Exercise 5.19. Repeat the previous argument for the other cases.

Exercise 5.20. How do you detect if the first or third qubits have been flipped?

Suppose that instead of a bit-flip error, the transmitted qubit undergoes an unexpected phase flip, that is,

$$|q\rangle = \alpha_0|0\rangle + \alpha_1|1\rangle \longmapsto \alpha_0|0\rangle - \alpha_1|1\rangle\,. \tag{5.61}$$

We would like to be able to correct these types of errors. For simplicity, suppose that the initial qubit is one of the Hadamard basis states, $|+\rangle$ or $|-\rangle$. A phase-flip error will then be

$$|\pm\rangle = \frac{1}{\sqrt{2}}\big(|0\rangle \pm |1\rangle\big) \longmapsto \frac{1}{\sqrt{2}}\big(|0\rangle \mp |1\rangle\big) = |\mp\rangle\,. \tag{5.62}$$

Thus, a phase flip $|\pm\rangle \mapsto |\mp\rangle$ is somehow equivalent to a bit flip $|i\rangle \mapsto |\bar{i}\rangle$. Since $X|i\rangle = |\bar{i}\rangle$ and $Z|i\rangle = (-1)^i Z|i\rangle$, and, on the other hand, $Z|\pm\rangle = |\mp\rangle$ and $X|\pm\rangle = \pm|\pm\rangle$, we conclude that we can use the previous argument of bit-flip errors to correct phase flips. We only need to change $Z\,Z\,I$ and $I\,Z\,Z$ by $X\,X\,I$ and $I\,X\,X$. Let us see this in more detail.

The three-qubit repetition code for phase-flip errors begins with the following logical computational qubits,

$$|0\rangle \longmapsto |0\rangle_L = |+\ +\ +\rangle\,, \qquad |1\rangle \longmapsto |1\rangle_L = |-\ -\ -\rangle\,. \tag{5.63}$$

The logical qubit is then,

$$\begin{aligned} |q\rangle_L &= \frac{1}{\sqrt{2}}\big(|+\ +\ +\rangle + |-\ -\ -\rangle\big) \\ &= \frac{1}{\sqrt{2}}\sum_i |(-1)^i\,(-1)^i\,(-1)^i\rangle\,. \end{aligned} \tag{5.64}$$

Exercise 5.21. Design the circuit that creates the logical computational qubits.

If by chance a phase flip occurs in (5.64), for one of the three qubits we will have $|\pm\rangle \mapsto |\mp\rangle$ or, equivalently, $|(-1)^i\rangle \mapsto |(-1)^{i+1}\rangle$. Note that this is completely analogous to a bit flip in (5.57). Thus, the phase-flip error syndrome will be detected applying the gates $X\,X\,I$ and $I\,X\,X$ (in whatever order you prefer because they commute). Once the error has been detected, the phase is restored by applying a Z gate to the appropriate qubit.

Exercise 5.22. Repeat everything we did above for the detection of bit-flip errors, but now for phase-flip errors. Be careful with the notation.

To conclude, let us see how the stabilizer formalism applies to Shor's nine-qubit code. This is a code that corrects any possible error that could occur to a single qubit, not only bit-flip or phase-flip errors. Since we have already shown that phase flips are equivalent to bit flips, for brevity, let us consider only the bit flip error. In this case, the logical computational qubits are prepared as follows,

$$|0\rangle \longmapsto |0\rangle_L = (|0\ 0\ 0\rangle + |1\ 1\ 1\rangle)^{\otimes 3}\,, \tag{5.65}$$

$$|1\rangle \longmapsto |1\rangle_L = (|0\ 0\ 0\rangle - |1\ 1\ 1\rangle)^{\otimes 3}\,. \tag{5.66}$$

Or, in more compact notation,

$$\begin{aligned} |i\rangle \longmapsto |i\rangle_L &= (|0\ 0\ 0\rangle + (-1)^i|1\ 1\ 1\rangle)^{\otimes 3} \\ &= \Big(\sum_i (-1)^{ij}|j\ j\ j\rangle\Big)^{\otimes 3} \\ &= \Big(\sum_i (-1)^{ij}|j\ j\ j\rangle\Big)\Big(\sum_i (-1)^{ik}|k\ k\ k\rangle\Big)\Big(\sum_i (-1)^{il}|l\ l\ l\rangle\Big)\,. \end{aligned} \tag{5.67}$$

Note that we have extended the space from $\mathcal{H}_q \cong \mathbb{C}^2$ to $\mathcal{H}_{q_9} \cong \mathbb{C}^{2^9} = \mathbb{C}^{512}$.

Exercise 5.23. Design the circuit that creates the logical computational qubits.

The logical qubit is then

$$|q\rangle_L = \sum_i \alpha_i \Big(\sum_i (-1)^{ij}|j\ j\ j\rangle\Big)\Big(\sum_i (-1)^{ik}|k\ k\ k\rangle\Big)\Big(\sum_i (-1)^{il}|l\ l\ l\rangle\Big)\,. \tag{5.68}$$

Suppose now that a bit flip occurs, for example,

$$|q\rangle_E = \sum_i \alpha_i \Big(\sum_i (-1)^{ij}|j\ \bar{j}\ j\rangle\Big)\Big(\sum_i (-1)^{ik}|k\ k\ k\rangle\Big)\Big(\sum_i (-1)^{il}|l\ l\ l\rangle\Big)\,. \tag{5.69}$$

We detect it simply by letting the qubit pass through a $(Z\,Z\,I)\,I^{\otimes 3}\,I^{\otimes 3}$ gate, take note of the eigenvalue (that is, if we measure plus or minus one) and then let it enter a $(I\,Z\,Z)\,I^{\otimes 3}\,I^{\otimes 3}$ gate. The procedure is the same as above. Errors in the other qubits are detected in a similar way.

Exercise 5.24. Write down the gates needed to detect other bit-flip errors.

For phase-flip errors, we use Hadamard gates to prepare the logical qubit in the state

$$|q\rangle_L = \sum_i \alpha_i \Big(\sum_i (-1)^{ij} |(-1)^j \ (-1)^j \ (-1)^j\rangle \Big)$$
$$\Big(\sum_i (-1)^{ik} |(-1)^k \ (-1)^k \ (-1)^k\rangle \Big) \Big(\sum_i (-1)^{il} |(-1)^l \ (-1)^l \ (-1)^l\rangle \Big) . \quad (5.70)$$

Error syndromes are detected by using gates made of products of Pauli gates such as, for example, $(X\,X\,I)\,I^{\otimes 3}\,I^{\otimes 3}$ and $(I\,X\,X)\,I^{\otimes 3}\,I^{\otimes 3}$. The error, if any, is corrected by applying the Pauli gate Z to the qubit whose phase was changed during the transmission. In our example, the gate is $(I\,Z\,I)\,I^{\otimes 3}\,I^{\otimes 3}$.

Exercise 5.25. Provide all the details to complete the above argument.

5.4 Fault-Tolerant QEC

Fault tolerance is based on the realistic projection that future quantum computers will not be perfect and errors will certainly occur during the computational process. The goal of fault-tolerance quantum computing is to develop a complete set of procedures to ensure that the results obtained by using imperfect quantum computers are nonetheless reliable. The quantum circuit must, then, be designed so that in case there is an error in the middle of the process, the error is promptly corrected and does not propagate uncontrollably to other components of the circuit. In other words, the goal of fault-tolerance quantum computing is to keep errors under control.

In the previous section we saw how the repetition code protects a single qubit from undesirable bit and phase flips. But, we have not discussed how the logical qubit, which is created to protect the single qubit during transmission, transforms when it enters a gate. These gates, as well, have to be designed so that they mitigate the errors and they do not propagate inside the circuit. Finally, the circuits built to correct errors are also prone to errors and these have to be corrected too.

This will be the subject of future notes.

6 Bibliography

In addition to the references given in our previous review paper, you may want to consult the free resources listed below. The various courses taught by Daniel Gottesman at Perimeter Institute are worth watching.

[1] S. Aaronson, "Introduction to Quantum Information Science II: Lecture Notes".

[2] Y. Cao et al., "Quantum Chemistry in the Age of Quantum Computing".

[3] A. Childs, "Lecture Notes on Quantum Algorithms".

[4] S. Devitt et al., "Quantum Error Correction for Beginners".

[5] D. Gottesman, "Stabilizer Codes and Quantum Error Correction".

[6] D. Lidar, "Lecture Notes on the Theory of Open Quantum Systems".

[7] S. McArdle et al., “Quantum Computational Chemistry”.

[8] R. O’Donnell & J. Wright, “Quantum Computation”.

[9] J. Tilly et al., “The Variational Quantum Eigensolver”.

[10] S. Wilde, “From Classical to Quantum Shannon Theory”.

[11] O. Zapata, “A Short Introduction to Quantum Computing for Physicists”.

Index

Additivity of the von Neumann entropy, 31
Ansatz (trial) state, 44

Bayes' theorem, 20
Binary probability distribution, 22
Bit (information unit), 20
Bit-flip channel, 50
Bit-phase-flip channel, 51
Bloch ball, 9
Bloch vector, 8

Channel (quantum), 49
Classical-quantum hybrid algorithm, 43
Conditional information content of two events, 23
Conditional mutual information, 26
Conditional relative entropy, 28
Conditional von Neumann entropy, 32

Density operator formalism, 5
Discrimination, 27
Divergence, 27

Fault tolerance, 59

Group generators, 56

Heuristic algorithms, 46

Independent systems, 19
Information content of a single event, 20

Joint information content of two events, 23
Joint probability, 19
Joint relative entropy, 28
Joint von Neumann entropy, 31

Kraus operator, 15

Marginal conditional entropy, 23
Marginal probability, 20
Mutual information of two probability distributions, 25
Mutual quantum information, 32

Phase angle, 35
Phase-flip channel, 51
Private (cryptographic) key, 46

Quantum channel, 13
Quantum cryptography, 46
Quantum key distribution (QKD), 46
Quantum map, 13

Reduced von Neumann entropy, 30
Relative entropy between two events, 27
Relative entropy between two probability distributions, 27
Relative von Neumann entropy, 32

Shannon conditional entropy, 24
Shannon conditional entropy of two events, 23
Shannon entropy of a probability distribution, 21
Shannon entropy of a single event, 20
Shannon joint entropy, 23
Shannon joint entropy of two events, 23
Shannon marginal joint Shannon entropy, 23
Stabilizer, 53
Stabilizer code, 53
Stabilizer group, 53
Stabilizer operator, 53
State operator, 7
Strong subadditivity, 26
Subadditivity, 31
Superoperator, 13

Trial (ansatz) state, 44

Variational algorithm, 43
von Neumann entropy, 28

An Introduction to Portfolio Optimization with Quantum Computers

Oswaldo Zapata[1]

Abstract

The purposes of this review article is to provide a self-contained introduction to one of the most promising applications of quantum computing to finance: portfolio optimization. In particular, we focus on the Quantum Approximate Optimization Algorithm (QAOA). These notes have been written so that anybody with an intermediate knowledge of linear algebra, calculus and quantum physics can read it. Moreover, tens of exercises have been included to help the reader fully grasp the subject matter.

1 Introduction **2**

2 Quantum Computing Review **3**
2.1 Main Concepts . . . 3
2.2 Computational Errors and Fault Tolerance . . . 7
2.3 Hybrid Quantum-Classical Algorithms . . . 9

3 Elements of Optimization Theory **10**
3.1 Continuous Optimization . . . 16
3.1.1 Unconstrained Problems . . . 16
3.1.2 Constrained Problems . . . 22
3.2 Dual Optimization Problems . . . 28
3.3 Integer Programs . . . 30

4 Classical Portfolio Optimization Theory **33**
4.1 Mathematical Description of an Investment Portfolio . . . 33
4.2 The Mean-Variance Model . . . 37
4.3 Portfolio Optimization as a Quadratic Programming . . . 47

5 Quantum Portfolio Optimization **51**
5.1 The Quantum Approximate Optimization Algorithm . . . 51
5.2 Portfolio Optimization via the QAOA . . . 51

6 Bibliography **58**

[1]Please send your feedback to zapata.oswaldo@gmail.com.

1 Introduction

The potential applications of quantum computing to finance are many and, more importantly, they seem to be just around the corner. Rightly so, in recent years a growing number of survey papers have been published reviewing the state of the field. These articles, though, have been written for the expert and are difficult to read for the uninitiated. The purpose of the present notes is precisely to provide to professional physicists and engineers, as well as students, with only some elementary knowledge of undergraduate mathematics and intermediate quantum physics, the necessary theoretical background to understand these review articles as well as to provide an entry point to the specialized research literature.

You may think that in order to apply quantum computing to financial problems, a sound knowledge of both subjects, quantum physic and finance, is required. This is certainly true if you want to have a broad and deep understanding of the subject, however, our goal here is more humble and we limit ourselves to basic results of quantum computing as applied to portfolio optimization. Even this simple purpose, though, requires a background knowledge in areas not usually covered in standard physics curricula. This is the reason why, in contrast to more advanced survey papers, we review optimization theory in some detail as well as the basics of modern portfolio theory.

The paper is organized as follows. We start in Section 2 with a quick overview of quantum computing. For a more complete introduction, check my two companion review articles in the Bibliography. In Section 3, we provide an introduction to optimization theory. Since this is a vast subject with many practical applications, we focus only on the concepts needed for portfolio optimization. In Section 4, we define a financial portfolio and explain what it means to optimize it. In Section 5, we put into practice everything learned so far and explain how quantum computers can, in principle, help optimize investment portfolios. It should be noted that Sections 2, 3 and 4, with the exception of Subsection 4.3, are independent from each other and can be read in any order.

2 Quantum Computing Review

In few words, *quantum computation* is the quantum mechanical way of solving a computational problem. That is, quantum computation uses the principles and mathematical formalism of quantum mechanics, such as state vectors, unitary operators and measurements, to arrive at the logical conclusion of a certain computational problem. A *quantum computer*, on the other hand, is the physical device that realizes the quantum computational process we are interested in. For decades it was thought that the only machines that could ever be able to perform quantum computations were quantum computers built exclusively with quantum components. In this section, we will see that this is, in fact, no more true. Nowadays, experts believe that the first machines that will improve upon classical super computers will be hybrid devices made of quantum and classical parts working in tandem.

This section is organized as follows. In Subsections 2.1 and 2.2, we review some basic concepts of quantum computing, in particular, the circuit model of quantum computation and the importance of error correction. Subsection 2.3 is meant as a preliminary introduction to the hybrid quantum-classical computational models mentioned above.

2.1 Main Concepts

I have already provided a rather exhaustive introduction to quantum computing in QC1 and QC2. If you find the following discussion too short, take a look at these papers or the corresponding literature. However, be assured that we will not need everything discussed there. The following review emphasizes only the main theoretical ideas of quantum computing and the practical difficulties scientists face when trying to implement these ideas in the laboratory.

Let us start with the most basic concepts. A *model of computation*, broadly speaking, is the logical way to proceed given some basic elements and instructions. More precisely, a model of computation is defined by a set of abstract objects and a set of elementary operations on these objects. An *algorithm* is a sequence of precise instructions within a model of computation, created specifically to solve a computational problem.

For example, the *Boolean* or *binary model of computation* is based on the so called *Boolean algebra*. In a Boolean algebra there are only two elements, conventionally denoted by 0 and 1, and three elementary operations, called NOT, AND and OR. If we denote any two arbitrary elements by $i, j = 0, 1$, and use the standard notation of the arithmetic system, the elementary operations are defined as follows,

$$\mathrm{NOT}\,(i) = \mathrm{NOT}\,i = \bar{i} = 1 - i\,, \tag{2.1}$$

$$\mathrm{AND}\,(i, j) = i\,\mathrm{AND}\,j = ij\,, \tag{2.2}$$

$$\mathrm{OR}\,(i, j) = i\,\mathrm{OR}\,j = i + j - ij\,. \tag{2.3}$$

In the context of computer science, an element i of the Boolean algebra is known as a *binary digit* or *bit*, for short. The elementary operations are called *Boolean logical gates*.

Exercise 2.1. Work out explicitly the action of each logical gate on the bits 0 and 1.

A *Boolean circuit* is a sequence of Boolean logical gates. As is usually the case for electric circuits, Boolean circuits are often represented visually by *circuit diagrams*. Since every Boolean circuit is a deterministic process (because the action of each individual gate is, in fact, deterministic), we have that for every string of bits that enters a Boolean circuit (the *input*), there is a unique string of bits that exists it (the *output*). Any Boolean circuit, hence, defines a unique *Boolean function*. The reverse problem is also interesting: given a Boolean function, what is the Boolean circuit that realizes it? In this case, though, it can be proved that the circuit is not unique. For instance, some circuits may contain more logical gates than others. A circuit that can solve a problem with the less logical gates than others is said to be more *efficient*.

The analysis of the depth (size) and complexity of a circuit, that is, the number of gates employed to performed a computation, is called *circuit complexity*. For example, it decides whether a problem can or cannot be solved by certain model of computation. In addition to studying theoretical problems, circuit complexity is also of mayor practical importance. Suppose that, for example, the NOT gate we have built sometimes gives the wrong result. That is, when the bit i enters our NOT gate, on the other side sometimes we read the same bit i instead of $\bar{i}$. This is known as a *computational error*. If we use many of these faulty NOT gates in a circuit, the risk is that the error propagates through the circuit, giving, possibly, a wrong computational result. Obviously, the deeper the circuit, the greater the probability that the final computational result will be wrong. It is, thus, crucial to know the level of error that an algorithm can tolerate. Since errors are unavoidable, it is obvious that this posed a serious concern in the earlier stages of our modern digital era.

Fortunately, by the mid-20th century, physicists have already discovered the electronic components needed to built reliable machines based on the binary circuit model of computation. They discovered that they could build large computational devices whose components were in one-to-one correspondence with the Boolean algebra. They are what we call today *digital computers* or, simply, *computers*. The solution of any computational problem was thus reduced to the following steps: 1. translate the computational problem to an equivalent Boolean function, 2. find the instructions, that is, the algorithm, that solves it, 3. wait for the machine to do its job. It was believed that, thanks to digital computers, the solution to any solvable problem was just a matter of time. The purpose of practical computer science was then to discover more efficient algorithms and more powerful computers.

The *quantum model of computation* is a completely different logical system. In contrast to the binary model of computation, the fundamental ideas of this model are based, as its name indicates, on the principles of quantum mechanics. Basically, in quantum computation, we do not deal with two different set of objects, say ones and zeros, but with their linear superposition. Instead of bits, we now have *qubits* (quantum binary digits). Moreover, the *quantum logical gates* are unitary transformations on qubits. A *quantum circuit* is a sequence of quantum gates connected by quantum channels by means of which the qubits are transferred. As in any quantum experiment, at the end of the quantum circuit, there is a measurement. According to the postulates of quantum mechanics, when a qubit passes through a quantum gate or circuit, in general, there is no certainty about the result of the measurement. In other words, the result is probabilistic and not deterministic as in the binary (classical) case. A *quantum algorithm* is a specific set of prescriptions, including

the initial qubit, the arrangement of gates and the appropriate measurements at the end, intended to solve a computational problem. All these concepts are at the heart of a contemporary scientific paradigm known as the *quantum circuit model of computation*. The purpose of this model, at least for those scientists interested in the physical applications of the theory, is the development of the machines that will implement this model. These devices are the so called *quantum computers*.

Let us see in more detail some of the basic components of the quantum circuit model of computation.

A *single qubit* is the simplest element of this model. If we denote by $|0\rangle$ and $|1\rangle$ the two classical states 0 and 1, any single qubit $|q\rangle$ will be a liner superposition of these states,

$$|q\rangle = \alpha_0 |0\rangle + \alpha_1 |1\rangle = \sum_{i=1}^{2} \alpha_i |i\rangle \ , \tag{2.4}$$

where α_0 and α_1 are complex numbers. The states $|0\rangle$ and $|1\rangle$ are called the *computational basis states*. Quantum mechanic affirms that the probability of measuring the single qubit $|q\rangle$ in state $|i\rangle$ is $|\alpha_i|^2$.

An n *qubit* is, in general, an entangled state of n single qubits. It is expressed as follows,

$$|Q\rangle = \sum_{i_1,\ldots,i_n} \alpha_{i_1\ldots i_n} |i_1 \ldots i_n\rangle \ , \tag{2.5}$$

where every bit $i_1, \ldots, i_n$ in the string $i_1 \ldots i_n$ takes the values 0 and 1 and the coefficients $\alpha_{i_1\ldots i_n}$'s are complex numbers. Note that some states of the n qubit are not entangled. For example, the following product states are non-entangled states,

$$|i \ldots i\rangle = |i\rangle^{\otimes n} \ . \tag{2.6}$$

A *quantum gate* is a unitary transformation on an input qubit. For instance, a *single-qubit gate* is a unitary transformation on a single qubit,

$$|q\rangle \longmapsto U |q\rangle = \sum_{i=1}^{2} \alpha_i U |i\rangle \ , \tag{2.7}$$

where, by definition of unit operator, $\langle q'|U^\dagger U|q\rangle = \langle q'|q\rangle$. Examples of such single-qubit gates are the *Pauli gates*,

$$X|0\rangle = |1\rangle \ , \qquad X|1\rangle = |0\rangle \ , \tag{2.8}$$

$$Y|0\rangle = i|1\rangle \ , \qquad Y|1\rangle = -i|0\rangle \ , \tag{2.9}$$

$$Z|0\rangle = |0\rangle \ , \qquad Z|1\rangle = -|1\rangle = e^{i\pi}|1\rangle \ . \tag{2.10}$$

Exercise 2.2. Is it clear for you why the X gate is the quantum analog of the classical NOT gate?

Exercise 2.3. Write in a compact form the action of the Pauli gates on general single qubits.

The Pauli gate Z is a special case, when $\phi = \pi$, of the *relative phase gate (phase shift gate)*,

$$P(\phi)|0\rangle = |0\rangle \ , \qquad P(\phi)|1\rangle = e^{i\phi}|1\rangle \ . \tag{2.11}$$

Each of the following operators, known as *rotation gates*, is a single-qubit rotation of θ_a radians about the a axis,

$$R_a(\theta_a) = \cos(\theta_a/2)I - i\sin(\theta_a/2)\sigma_a\,, \tag{2.12}$$

where I is the identity operator and σ_a, with $a = x, y, z$, is another notation for the Pauli gates ($\sigma_x = X$, $\sigma_y = Y$ and $\sigma_z = Z$).

Exercise 2.4. Under which physical conditions is correct to say that $R_x(\pi) = X$?

In general, for an *n-qubit gate* acting on an n qubit,

$$|Q\rangle \longmapsto U\,|Q\rangle = \sum_{i_1,\ldots,i_n} \alpha_{i_1\ldots i_n} U\,|i_1\ldots i_n\rangle\ , \tag{2.13}$$

with $UU^\dagger = I_n$.

The *CNOT gate* is an example of a two-qubit gate,

$$\mathrm{CNOT}\,|i\ j\rangle = |i\ j\oplus i\rangle\ , \tag{2.14}$$

where the symbol $\oplus$ denotes the binary sum, $j \oplus i = (j+i)\,\mathrm{mod}2$.

Exercise 2.5. How does a CNOT gate act on an arbitrary two-qubit?

Any unitary transformation, it can be shown, can be approximated by a finite quantum circuit (that is, a circuit made of a finite number of quantum gates),

$$|Q\rangle \longmapsto U_C\,|Q\rangle \approx U_N\ldots U_1\,|Q\rangle\ . \tag{2.15}$$

Furthermore, just as any classical circuit can be decomposed into a sequence of NOT, AND and OR gates, any quantum circuit can be thought of as a composition of a finite number of elementary quantum gates. Any such finite set of gates is called a *set of universal quantum gates*. For example, the rotation gates, the phase shift gate and the CNOT gate, together form a universal set of quantum gates.

In quantum mechanics, any physical *measurement* is associated to a Hermitian operator called an observable, $M = M^\dagger$. Because the Pauli operators, as well their tensor products, are Hermitian,

$$\sigma_a = \sigma_a^\dagger \tag{2.16}$$

$$\sigma_{a_1}\ldots\sigma_{a_N} = (\sigma_{a_1}\ldots\sigma_{a_N})^\dagger\,, \tag{2.17}$$

they are often used to express any measurement performed on a quantum circuit. Since, in general, a measurement has an unpredictable effect on a quantum system (in contrast to classical systems where the effects of a measurement are insignificant), the measurement is an integral part of the quantum computational process. This is why we must always specify the measurements to be performed at the end of a quantum circuit. The outcomes are then probabilistic instead of deterministic.

After this quick survey of both the binary and quantum models of computation, you may be asking yourself: why do we need quantum computers if we already have classical (digital) computers which have been shown to be extremely reliable for solving most practical computational problems?

There are two main reasons to believe that, in some instances, quantum computers will supersede classical computers. The first reason is that quantum computers may solve problems much faster than classical computers. This is what is meant when it is said that quantum computers will be more "efficient" or "powerful" than classical computers. In practical terms, this means that, eventually, quantum computers will be much smaller (less complex) than classical computers built with the same computational purpose in mind. The second rationale behind the growing interest in quantum computing is that quantum computers will probably (there is no formal proof of it) be able to solve computational problems that classical computers are incapable to solve. That is, scientists expect them to solve problems that even the most powerful digital computers we can imagine cannot solve.

2.2 Computational Errors and Fault Tolerance

It is expected that future quantum computers, at least in the earlier stages of development, will not work exactly as described above. In other words, they will be prone to *computational errors*. The control of the propagation of these errors is a critical task in the progress toward a reliable quantum computer.

At the beginning of the modern digital era, digital components were also far from perfect and there was a need for error correction, that is, methods for detecting and correcting errors in the computational process. Nowadays, however, electronic components are so accurate that the probability of a computational error is insignificant and so there is no need for error correction.

One of the reasons why quantum circuit components are so unreliable is that it is very difficult to isolate them from their environments. In fact, these interactions are so drastic that they destroy the quantum mechanical behavior of the circuit elements. This effect, very well-known in quantum mechanics, is known as *decoherence*. This destructive effect of the environment has compelled physicists and engineers to develop new techniques to mitigate the influence of external factors on circuit components. On the other hand, theoretical physicists and computer scientists have invented procedures to detect and correct the various types of errors that may occur to these "noisy" quantum computers.

As an example of a concrete way experts have created to cope with decoherence, consider the *bit-flip error* that may occur in the propagation of a qubit from one gate to the other.

In the classical case, a bit flip is the only possible error, $i \to \bar{i}$. In order to protect it, we copy it and send several identical copies of it. That is, instead of i, we send $i\,i\,i$. If one of the bits is flipped, for example, if we receive $i\,\bar{i}\,i$ instead of $i\,i\,i$, we measure the three bits, detect that the second has been flipped and then correct it. The whole process can be summarized as follows,

$$i \xrightarrow{encode} i\,i\,i \xrightarrow{send\ (error\ occurs)} i\,\bar{i}\,i \xrightarrow{detect} \xrightarrow{correct} i\,i\,i \xrightarrow{decode} i \tag{2.18}$$

Note that this method would not work if several errors could occur simultaneously. In fact, the *majority voting* strategy we just used will "correct" the wrong bit. For example,

$$i \xrightarrow{encode} i\,i\,i \xrightarrow{send\ (errors\ occur)} i\,\bar{i}\,\bar{i} \xrightarrow{detect} \xrightarrow{correct} \bar{i}\,\bar{i}\,\bar{i} \xrightarrow{decode} \bar{i} \tag{2.19}$$

By repeating the initial bit many more times and assuming that the probability that an error occurs is very small, this correction procedure is completely reliable.

The *bit-flip quantum repetition code* is similar. However, there are some fundamental differences. Similar to the classical case, instead of sending one single qubit $|q\rangle$, we send a three qubit $|q\rangle_L$,

$$|q\rangle = \sum_{i=1}^{2} \alpha_i |i\rangle \xrightarrow{encode} |q\rangle_L = \sum_{i=1}^{2} \alpha_i |i\, i\, i\rangle \xrightarrow{send} \tag{2.20}$$

To distinguish $|q\rangle_L$ from the the initial qubit $|q\rangle$, the former is called the *logical qubit* and each of the state vectors in $|i\rangle\,|i\rangle\,|i\rangle$ are the *physical qubits*. Now, in the process of transmitting the qubit, errors may occur. In our example, suppose that it is a bit flip. For the initial single qubit $|q\rangle$ this would have meant

$$|q\rangle = \sum_{i=1}^{2} \alpha_i |i\rangle \longrightarrow \sum_{i=1}^{2} \alpha_i |\bar{i}\rangle \ , \tag{2.21}$$

however, since we are sending a logical three qubit, the error can now occur in any of the three physical qubits. Suppose that it is the second physical qubit which is flipped,

$$|q\rangle = \sum_{i=1}^{2} \alpha_i |i\rangle \xrightarrow{encode} |q\rangle_L = \sum_{i=1}^{2} \alpha_i |i\, i\, i\rangle \xrightarrow{send\ (error\ occurs)} \sum_{i=1}^{2} \alpha_i |i\, \bar{i}\, i\rangle \tag{2.22}$$

The detection and correction of an error in a qubit is not as simple as in the classical case. In fact, we can measure a classical bit an leave it almost undisturbed. However, according to the principles of quantum mechanic, the measurement of a qubit will project it along one of the observable states. To avoid this, we perform a *parity check*. I already explained how to do this in QC1, Subsection 5.2, and I will not repeat it here. After the identification of the error, we correct it and recover the initial qubit. The entire process can be summarized as follows,

$$|q\rangle = \sum_{i=1}^{2} \alpha_i |i\rangle \xrightarrow{encode} |q\rangle_L = \sum_{i=1}^{2} \alpha_i |i\, i\, i\rangle \xrightarrow{send\ (error\ occurs)} \sum_{i=1}^{2} \alpha_i |i\, \bar{i}\, i\rangle$$

$$\xrightarrow{parity\ check} \xrightarrow{correct} |q\rangle_L = \sum_{i=1}^{2} \alpha_i |i\, i\, i\rangle \xrightarrow{decode} \sum_{i=1}^{2} \alpha_i |i\rangle = |q\rangle$$

Other types of errors can occur and similar procedures have been created to detect and correct them. In addition to these errors related to the transmission of qubits, gates can also produce errors. Imagine, for example, that you expect a qubit $|i\rangle$ to exit a gate but instead the qubit $|\bar{i}\rangle$ exists it. What is worse, a qubit that is transferred through a noisy channel can enter a faulty gate. If the two errors add up, the final qubit could be unrecognizable. Thus, if we do not pay attention, the errors can propagate throughout the circuit, making the final computation unusable.

Finally, since the error correction process, that is, the encoding, detection and correction of errors, is done also by quantum devices, the latter will inevitably create additional errors. Error correction, hence, implies larger quantum computers and, thus, a higher probability that the computational result will be incorrect. However,

it has been proven that, under certain conditions, correction codes can make the computational error as small as desired. A *fault-tolerant quantum computer* is a quantum computer built such that the errors occurring in the logical qubits at every stage of the process are corrected along the way so that the final computational result is reliable. This, however, is still too far out in the future.

2.3 Hybrid Quantum-Classical Algorithms

For years it was thought that quantum computers had to be built exclusively with quantum components. This seemed obvious: given that we wanted to demonstrate the superiority of quantum over classical computation, the quantum device had to be purely quantum. The main obstacles in front of this ideal goal were, and still are, the development of new hardware (less noisy) and the invention of the appropriate quantum error correcting codes. This future situation is known as the *fault-tolerant quantum era*.

However, in recent years, scientists have accepted that fault-tolerant quantum computers will not be available any time soon. They started, then, to look for more realistic algorithms that could be implemented in near-term quantum computers, characterized by a moderate amount of noise and a relatively small number of qubits and gates. This is the moment we are living right now and is called the *Noisy-Intermediate Scale Quantum era (NISQ era)*. According to the experts, we will stay several years (even decades) in this stage of development of the quantum technology before reaching fault tolerance.

The algorithms they expect to be implemented in the near term are the so called *hybrid quantum-classical algorithms*. They combine a quantum and a classical part. The quantum part tackles a problem that has been proven to be hard for classical computers while, on the other hand, the classical computer performs an easy task.

The *variational quantum algorithms (VQAs)* we will discuss in these notes are NISQ algorithms, that is, hybrid quantum-classical algorithms devised to prove *quantum advantage* (practical *quantum supremacy*) in the near future. Since many problems, not only in physics and chemistry but also, as we will see in the last section, in finance, have a common basic structure, the techniques proper to VQAs can be applied to a wide variety of situations.

It is common to hear that some variational quantum algorithms, in particular the quantum approximate optimization algorithm (QAOQ) we will present below, are *heuristic*. By this it is meant that even though there is so far no rigorous proof that they are more efficient than the known classical algorithms, there are good theoretical reasons to be optimistic. The promise is that future results may show their advantage. The implementation of these types of algorithms, though, may turn out to be trickier than expected by some of the most enthusiastic supporters of these models.

With this brief introduction to quantum computing, let us now cover everything we need to know about optimization theory to fully understand the QAOA.

3 Elements of Optimization Theory

Optimization theory is a vast topic with many applications in pure science, engineering and finance. In this section, however, we content ourselves with a brief introductory discussion of the main concepts and techniques needed for the optimization of financial portfolios.

The basic idea of optimization is easy to understand. Suppose that you are given a real-valued function that depends on a set of parameters and you are asked to find the values of the parameters that minimize the given quantity. That is, provided a real-valued multivariable function, the goal is to find the values of the independent variables that correspond to the minimum, or maximum, of the function (we will discuss below the possibility of multiple minimums and maximums). Sometimes, the independent variables are subject to additional conditions, making the problem even more difficult. Why the function to be minimized depends on the variables the way it does or why the variables are restricted to some set of values and not others, has to do with the particular problem we want to solve. Because of the complexity and diversity of the problems, different methods have been developed to tackle different sorts of optimization problems.

We start with the simplest case, the optimization of a differentiable real-valued function on a single variable, something taught in every elementary calculus course. Remember that the points where the first derivatives vanish are the critical points and the second derivatives determine whether these points are minimizers, maximizers or points of inflection. In principle, this is all we need to optimize a differentiable real-valued function on a single variable. However, for algebraic reasons, in most cases the solutions are difficult if not impossible to compute exactly. This is the reason why numerical methods and computer programs are often necessary. Something similar, but more difficult, happens with multivariable real-valued functions. The formulation of an optimization problem is relatively simple, the real difficulty is solving it.

Let us begin by reminding some basic definitions. Given the differentiable function $f\colon I \subseteq \mathbb{R} \to \mathbb{R}$, $x \mapsto f(x)$, where I is an open subset of the real line, a *critical point* x_c of f is a point in I where

$$\frac{\partial f(x_c)}{\partial x} = 0\,. \tag{3.1}$$

There are two types of critical points: *optimizers*, that we denote by x^*, and non-optimizers. Additionally, an optimizer can be a *minimizer*, x_{min}, or a *maximizer*, x_{max}. These are the points where the function attains a minimum or a maximum, respectively. To determine whether a critical point is a minimizer or a maximizer, we use the *second derivative test*. It states that, if the second derivative at a critical point is positive, then the critical point is a minimizer,

$$\frac{\partial^2 f(x_{min})}{\partial^2 x} > 0\,, \tag{3.2}$$

and, if it is negative, then it is a maximizer,

$$\frac{\partial^2 f(x_{max})}{\partial^2 x} < 0\,. \tag{3.3}$$

Non-optimizers, that is, critical points that are not minimizers or maximizers, are called *inflection points*.

Exercise 3.1. Show that at an inflection point the graph of a function changes concavity.

Exercise 3.2. Find the critical points of the functions x^2, x^3 and x^4. Can the second derivative test be used to determine the nature of their critical points?

Exercise 3.3. Prove the second derivative test using *Taylor's formula*,

$$f(x) = \sum_{k=0}^{n-1} \frac{1}{k!} \frac{d^k f(a)}{dx^k} (x-a)^k + \frac{1}{n!} \frac{d^n f(x')}{dx^n} (x-a)^n \,, \tag{3.4}$$

where x' is a point strictly between a and x.

Exercise 3.4. If a function has multiple extrema (an extremum can be a minimum or a maximum and the plural of extremum is *extrema*), how do you distinguish between local and global minimums and maximums?

Much more could be said about the optimization of real-valued functions of a single variable. For example, we could be interested in a function defined on a closed subset of the real line, in which case we had to consider the values taken by the function at the end-points. Note that, the previous theory can be applied to any differentiable function, however, in many practical situations the function is not differentiable and other methods have to be used.

These simple mathematical concepts find many uses in the physical sciences. For example, as we all learned in our first physics course, we can determine the highest altitude reached by an object thrown upward. We can, as well, find the distance at which the potential energy of an object attached to a spring is the minimum. We also learned, however, that one-dimensional problems are not always so easy to solve. For instance, in many occasions it is impossible to predict the exact position of an object moving under certain forces. In addition to one-dimensional problems, two-dimensional optimization problems are also common in elementary physics. For example, the calculation of the optimal angle a projectile has to be launched in order to reach the maximum horizontal distance. Another classic optimization problem of several variables is the construction of a cylindrical package with a fixed volume and using the least amount of material.

Exercise 3.5. i) Find the highest altitude reached by a particle thrown upward with velocity v_0. ii) What is the stretching distance corresponding to a minimum of the potential energy of a spring (assuming Hooke's law)?

Exercise 3.6. State formally all the examples mentioned in the previous paragraph and solve them.

In the jargon of optimization theory, the problem of finding the highest altitude reached by an object thrown upward is stated as follows. Given the *objective function*,

$$h(t) = v_0 t - \frac{1}{2} g t^2 \,, \tag{3.5}$$

where t is a real variable and v_0 and g are positive constants, maximize the function $h(t)$,

$$\max_t h(t) \,. \tag{3.6}$$

If it is clear what the independent variable is, in this case t, we simply write

$$\max\, h(t)\,. \tag{3.7}$$

The independent variable is usually called the *decision variable* and the constants (v_0 and g in our problem) are the *parameters*. Our goal, then, is to find the time (or times) t^* when $h(t^*) \geq h(t)$ for all $t, t^* \in \mathbb{R}$. Since the objective function $h(t)$ is such that $h(t^*) = h(t)$ only when $t = t^*$, the solution is unique. It is easy to show that the critical point is $t_c = v_0/g$ and it is a maximizer, $t_c = t_{max} = v_0/g$. The highest altitude is, $h(t_{max}) = h_{max} = v_0^2/2g$.

Exercise 3.7. Fill in the gaps in the calculations above.

The problem of finding the optimal launch angle is formulated as follows. The objective function is the horizontal distance x, which is a real-valued function in two decision variables (time t and the initial angle θ),

$$\max_{t,\theta}\, x(t,\theta)\,. \tag{3.8}$$

From elementary kinematics, we know that this function is given by

$$x(t,\theta) = v_0 t\cos\theta\,, \tag{3.9}$$

where v_0 is a positive constant (the initial speed of the object). The value of x we are interested in is the horizontal distance at ground level, therefore, we must add the following *equality constraint*

$$v_0 t\sin\theta - \frac{1}{2}\,gt^2 = t\Big(v_0\sin\theta - \frac{1}{2}\,gt\Big) = 0\,. \tag{3.10}$$

If we assume that the projectile is launched at $t = 0$, then, we must also include the *inequality constraint* $t > 0$. These two conditions can be collected in one single equality constraint,

$$h(t,\theta) = v_0\sin\theta - \frac{1}{2}\,gt = 0\,. \tag{3.11}$$

The problem we have to solve is, thus, the max $x(t,\theta)$ subject to (3.11). So far we have not mentioned the condition on the launch angle. For physical reasons we impose $0 < \theta < \pi/2$, a condition which is consistent with (3.11). This is a problem we know how to solve: we first find t using the constraint (3.11) and substitute it in (3.9), we then differentiate with respect to θ, equalize to zero, and finally determine the angle θ that solves the equation. This gives $\theta_c = 45°$. The second derivative test confirms that this is indeed a maximizer, $\theta_{max} = 45°$.

Exercise 3.8. Compute explicitly all the steps indicated above.

This high-school problem can also be solved using the *Lagrange multipliers method*. Here we think of x as a scalar field in the t-θ plane. Denoting by $(t,\theta)^* = (t^*,\theta^*)$ the pair of points in the t-θ plane that maximizes the distance function (3.9) and, at the same time, satisfies the constraint (3.11), the Lagrange multipliers method states that

$$\frac{\partial_\theta x(t,\theta)^*}{\partial_t x(t,\theta)^*} = \frac{\partial_\theta h(t,\theta)^*}{\partial_t h(t,\theta)^*}\,. \tag{3.12}$$

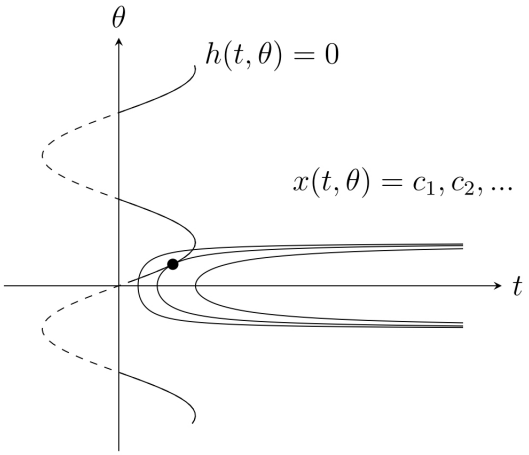

Fig. 1. Height and level curves of the position scalar field.

Geometrically, the left hand side represents the tangent to the level curve of $x(t,\theta)$ at the optimizer, and the right hand side is the tangent to the constraint curve $h(t,\theta)$ at that same point. Another way of writing the previous relation is,

$$\frac{\partial_t x(t,\theta)^*}{\partial_t h(t,\theta)^*} = \frac{\partial_\theta x(t,\theta)^*}{\partial_\theta h(t,\theta)^*} = \mu^* \,. \tag{3.13}$$

where the constant μ^* is known as a *Lagrange multiplier*. We obtain two independent equations out of this relation,

$$\partial_t x(t,\theta)^* - \mu^* \, \partial_t h(t,\theta)^* = 0 \,, \tag{3.14}$$

$$\partial_\theta x(t,\theta)^* - \mu^* \, \partial_\theta h(t,\theta)^* = 0 \,. \tag{3.15}$$

These equations plus the equality constraint,

$$h(t,\theta)^* = 0 \,, \tag{3.16}$$

is the system of equations that we now have to solve. Thus, in the current example, the Lagrange multipliers method replaces the problem of maximizing a function in two variables, $x(t,\theta)$, with one constraint $h(t,\theta)$, with the problem of solving a system of three equations with three unknowns, t^*, θ^* and μ^*.

Exercise 3.9. Solve the launch angle optimization problem using the Lagrange multipliers method.

The Lagrange multipliers method can be restated as follows. We first construct the *Lagrangian function*,

$$L(t,\theta,\mu) = x(t,\theta) - \mu h(t,\theta) \,, \tag{3.17}$$

and then declare that the maximization problem is equivalent to solving the following system of equations,

$$\partial_t L(t,\theta,\mu)^* = 0 \,, \qquad \partial_\theta L(t,\theta,\mu)^* = 0 \,, \qquad \partial_\mu L(t,\theta,\mu)^* = 0 \,, \tag{3.18}$$

where $(t,\theta,\mu)^* = (t^*,\theta^*,\mu^*)$. Or, in more compact notation,

$$\nabla_{t,\theta} L(t,\theta,\mu)^* = \mathbf{0} \,, \qquad \partial_\mu L(t,\theta,\mu)^* = 0 \,. \tag{3.19}$$

Using (3.17), we then get that

$$\nabla_{t,\theta} x(t,\theta)^* - \mu^* \nabla_{t,\theta} h(t,\theta)^* = \mathbf{0}\,, \tag{3.20}$$

that is,

$$\nabla_{t,\theta} x(t,\theta)^* = \mu^* \nabla_{t,\theta} h(t,\theta)^*\,. \tag{3.21}$$

The geometric interpretation of this relation is that, at the optimizer, the gradient of the level curve of the objective function is proportional to the gradient of the constraint function.

Exercise 3.10. Draw the corresponding vectors in Figure 1.

The cylindrical package problem can be formulated in a similar way. The area of the package is now the objective function,

$$A(r,h) = 2\pi r^2 + 2\pi r h\,, \tag{3.22}$$

and the goal is to minimize it,

$$\min A(r,h)\,, \tag{3.23}$$

subject to the condition that the volume contained within the package is constant, that is, r and h are such that

$$V(r,h) = \pi r^2 h = V_0\,. \tag{3.24}$$

This is the only constraint of the problem (in addition to the fact that distances, areas and volumes are positive quantities). You can easily show that the cylinder we are looking for has $r_{min} = h_{min}/2 = \sqrt[3]{V_0/2\pi}$. First, use the equality constraint to express the area in terms of a single variable, then differentiate with respect to this variable and equalize to zero, finally solve for the variable and use again the constraint to find the solution. The second derivative test gives that it is a minimizer.

Exercise 3.11. Solve the problem by first considering $A(r)$ and then $A(h)$. In both cases, of course, you should get the same answer.

The Lagrange multipliers method starts with the Lagrangian function,

$$\begin{aligned} L(r,h,\mu) &= A(r,h) - \mu\big[V(r,h) - V_0\big] \\ &= 2\pi r^2 + 2\pi r h - \mu(\pi r^2 h - V_0)\,, \end{aligned} \tag{3.25}$$

and states that the solution is given by the system of equations

$$\nabla_{r,h} L(r,h,\mu)^* = \mathbf{0}\,, \qquad \partial_\mu L(r,h,\mu)^* = 0\,. \tag{3.26}$$

Exercise 3.12. Solve the cylindrical package optimization problem using the Lagrange multipliers method. Plot the level curves of the objective function and indicate the point where the constraint function is satisfied. Draw the gradients at the minimum point.

In addition to the volume constraint on the volume, we also mentioned that $r, h > 0$. However, we could have imposed a different set of inequality constraints,

$$r_a \le r \le r_b\,, \qquad h_c \le h \le h_d\,. \tag{3.27}$$

Exercise 3.13. What is the interpretation of this in the context of Exercise 3.12?

Now, suppose we want to minimize the same area function (3.22), but instead of the equality constraint on the volume, we have an inequality constraint,

$$V(r,h) \leq V_0\,. \tag{3.28}$$

The solution to this problem is different from the one given above. Again, we start with a Lagrangian function,

$$L(r,h,\lambda) = A(r,h) - \lambda\big[V(r,h) - V_0\big]\,. \tag{3.29}$$

Note that the Lagrange multiplier is now denoted by λ. The Lagrange multipliers method affirms that the solution to the minimization problem is given by the following system of equations,

$$\partial_r L(r,h,\lambda)^* = 0\,, \qquad \partial_h L(r,h,\lambda)^* = 0\,, \qquad \partial_\lambda L(r,h,\lambda)^* \leq 0\,. \tag{3.30}$$

The last relation implies that

$$\partial_\lambda L(r,h,\lambda)^* = V(r,h)^* - V_0 \leq 0\,. \tag{3.31}$$

The method states that, in addition to these equations, we must add the following conditions on the optimum value of the Lagrange multiplier,

$$\lambda^* \geq 0\,, \tag{3.32}$$

and

$$\lambda^*\big[V(r,h)^* - V_0\big] = 0\,. \tag{3.33}$$

To illustrate how the method works, suppose that we are only interested in the minimization of the side area of the cylindrical package, $a(r,h) = 2\pi rh$. That is,

$$\min\, a(r,h) = \min\, 2\pi rh\,, \tag{3.34}$$

The constraint on the decision variables is the following,

$$g(r,h) = r^2 + h^2 \leq R^2\,. \tag{3.35}$$

Exercise 3.14. Draw a picture explaining the problem.

According to (3.29), the Lagrangian of the problem is

$$L(r,h,\lambda) = 2\pi rh - \lambda(r^2 + h^2 - R^2)\,. \tag{3.36}$$

The system of equations to be solved is then,

$$2\pi h^* - 2\lambda^* r^* = 0\,, \quad 2\pi r^* - 2\lambda^* h^* = 0\,, \quad r^{*2} + h^{*2} - R^2 \leq 0\,, \tag{3.37}$$

with

$$\lambda^* \geq 0\,, \qquad \lambda^*(r^{*2} + h^{*2} - R^2) = 0\,. \tag{3.38}$$

From the first two equations we get,

$$\lambda^* = \frac{\pi h^*}{r^*}\,, \qquad \lambda^* = \frac{\pi r^*}{h^*}\,. \tag{3.39}$$

This gives, $r^{*2} = h^{*2}$. If we discard the solution $\lambda^* = 0$, because that would imply $r^* = h^* = 0$, the third equation in (3.37) gives $r^{*2} = h^{*2} = R^2/2$. Given that the Lagrange multiplier must be positive, there are two possible mathematical solutions to the problem,

$$r^* = \pm\sqrt{\frac{R^2}{2}}\,, \qquad h^* = \pm\sqrt{\frac{R^2}{2}}\,, \qquad \lambda^* = \pi\,. \tag{3.40}$$

Finally, since distances are positive, the correct physical solution is,

$$r^* = \frac{R}{\sqrt{2}}\,, \qquad h^* = \frac{R}{\sqrt{2}}\,, \qquad \lambda^* = \pi\,. \tag{3.41}$$

3.1 Continuous Optimization

A *continuous optimization problem* is an optimization problem with objective function defined on a continuous set. For example, all the problems discussed above are continuous because the decision variables were assumed to be continuous (distances, angles, etc). Furthermore, it is convenient to classify continuous problems into unconstrained and constrained problems. In particular, because of their accessibility, we will introduce linear and quadratic problems.

Since here I assume that you have already taken a formal calculus course, below you will not find rigorous definitions and proofs. For a detailed presentation, I recommend you consult the appropriate literature.

3.1.1 Unconstrained Problems

An *unconstrained (continuous) optimization problem* is an optimization problem where the only conditions on the decision variables are those imposed by the (continuous) objective function itself. For example, given a differentiable real-valued function f on an open subset $U \subseteq \mathbb{R}^n$,

$$f\colon U \subseteq \mathbb{R}^n \to \mathbb{R}\,, \qquad (x_1, \ldots, x_n) \mapsto f(x_1, \ldots, x_n)\,, \tag{3.42}$$

an *unconstrained (continuous) minimization problem*,

$$\min_{(x_1, \ldots, x_n)} f(x_1, \ldots, x_n)\,, \tag{3.43}$$

consists in finding the point $(x_1, \ldots, x_n)^* \in U$, for which

$$f(x_1, \ldots, x_n)^* < f(x_1, \ldots, x_n)\,, \tag{3.44}$$

for all $(x_1, \ldots, x_n) \in U - \{(x_1, \ldots, x_n)^*\}$. A similar definition applies for a maximization problem.

For objective functions depending on a small number of decision variables, standard calculus techniques are taught in elementary courses. We begin with the simplest multivariable example: a real-valued function in two real variables,

$$f\colon U \subseteq \mathbb{R}^2 \to \mathbb{R}\,, \qquad (x, y) \mapsto f(x, y)\,. \tag{3.45}$$

A *critical point* of f is a point $(x, y)_c \in U$ that satisfies,

$$\frac{\partial f(x, y)_c}{\partial x} = 0\,, \qquad \frac{\partial f(x, y)_c}{\partial y} = 0\,. \tag{3.46}$$

Of course, there are functions with multiple critical points. A critical point can be a minimizer, a maximizer or a saddle point (the equivalent of an inflection point for functions of a single variable). Minimizers and maximizers are both *optimizers*.

Exercise 3.15. Write down everything you remember about minimizers, maximizers and saddle points.

To determine whether a critical point of the function f is a minimizer, a maximizer or a saddle point, we introduce the so called *Hessian matrix*,

$$Hf(x,y) = \begin{bmatrix} \dfrac{\partial^2 f(x,y)}{\partial x^2} & \dfrac{\partial^2 f(x,y)}{\partial x \partial y} \\ \dfrac{\partial^2 f(x,y)}{\partial y \partial x} & \dfrac{\partial^2 f(x,y)}{\partial y^2} \end{bmatrix} . \tag{3.47}$$

Note that the Hessian matrix is symmetric if we assume that partial derivatives in U commute. The determinant of the Hessian matrix is known as the *Hessian*,

$$\det\big(Hf(x,y)\big) = \frac{\partial^2 f(x,y)}{\partial x^2}\,\frac{\partial^2 f(x,y)}{\partial y^2} - \left(\frac{\partial^2 f(x,y)}{\partial x \partial y}\right)^2 . \tag{3.48}$$

The nature of a critical point is decided by the *second derivative test*. According to it, a *minimizer* $(x,y)_{min}$ satisfies,

$$\frac{\partial^2 f(x,y)_{min}}{\partial x^2} > 0\,, \tag{3.49}$$

and

$$\det\big(Hf(x,y)_{min}\big) > 0\,. \tag{3.50}$$

For a *maximizer* $(x,y)_{max}$, on the other hand,

$$\frac{\partial^2 f(x,y)_{max}}{\partial x^2} < 0\,, \tag{3.51}$$

and

$$\det\big(Hf(x,y)_{max}\big) > 0\,. \tag{3.52}$$

A critical point which is not a minimizer or a maximizer is a *saddle point*.

Exercise 3.16. Use *Taylor's formula* for functions of several variables,

$$f(\mathbf{x}) = f(\mathbf{a}) + \nabla f(\mathbf{a}) \cdot (\mathbf{x}-\mathbf{a}) + \frac{1}{2}\,(\mathbf{x}-\mathbf{a}) \cdot Hf(\mathbf{x}')(\mathbf{x}-\mathbf{a}) + \dots\,, \tag{3.53}$$

where $\mathbf{x}'$ is a point strictly between $\mathbf{a}$ and $\mathbf{x}$, that is, $\mathbf{x}' = \mathbf{a} + t(\mathbf{x}-\mathbf{a})$ for $0 < t < 1$, to verify by yourself that the above is the correct criterion to classify critical points.

Exercise 3.17. Find and classify all the critical points of the function

$$f(x,y) = \frac{1}{2}\,a_{xx}x^2 + \frac{1}{2}\,a_{yy}y^2 + a_{xy}xy - a_x x - a_y y\,. \tag{3.54}$$

Exercise 3.18. Provide some examples of real-valued functions $f(x,y)$ for which the critical points are difficult to find. What about the classification of these points?

The next simplest example is a differentiable real-valued function in three variables,

$$f\colon U \subseteq \mathbb{R}^3 \to \mathbb{R}\,, \qquad (x,y,z) \mapsto f(x,y,z)\,. \tag{3.55}$$

A *critical point* $(x,y,z)_c$ satisfies

$$\frac{\partial f(x,y,z)_c}{\partial x} = 0\,, \qquad \frac{\partial f(x,y,z)_c}{\partial y} = 0\,, \qquad \frac{\partial f(x,y,z)_c}{\partial z} = 0\,. \tag{3.56}$$

The *Hessian matrix* in this case is,

$$Hf(x,y,z) = \begin{bmatrix} \dfrac{\partial^2 f(x,y,z)}{\partial x^2} & \dfrac{\partial^2 f(x,y,z)}{\partial x \partial y} & \dfrac{\partial^2 f(x,y,z)}{\partial x \partial z} \\ \dfrac{\partial^2 f(x,y,z)}{\partial y \partial x} & \dfrac{\partial^2 f(x,y,z)}{\partial y^2} & \dfrac{\partial^2 f(x,y,z)}{\partial y \partial z} \\ \dfrac{\partial^2 f(x,y,z)}{\partial z \partial x} & \dfrac{\partial^2 f(x,y,z)}{\partial z \partial y} & \dfrac{\partial^2 f(x,y,z)}{\partial z^2} \end{bmatrix}. \tag{3.57}$$

It is symmetric if we assume that the partial derivatives commute. The second derivative test is similar to that stated above for functions of two variables. Consider the following determinants at a critical point,

$$\Delta_1(x,y,z)_c = \det\left(\frac{\partial^2 f(x,y,z)_c}{\partial x^2}\right), \tag{3.58}$$

$$\Delta_2(x,y,z)_c = \det\begin{bmatrix} \dfrac{\partial^2 f(x,y,z)_c}{\partial x^2} & \dfrac{\partial^2 f(x,y,z)_c}{\partial x \partial y} \\ \dfrac{\partial^2 f(x,y,z)_c}{\partial y \partial x} & \dfrac{\partial^2 f(x,y,z)_c}{\partial y^2} \end{bmatrix}, \tag{3.59}$$

$$\Delta_3(x,y,z)_c = \det\big(Hf(x,y,z)_c\big)\,. \tag{3.60}$$

The *second derivative test* affirms that at a *minimizer* $(x,y,z)_{min}$,

$$\Delta_1(x,y,z)_{min} > 0\,, \qquad \Delta_2(x,y,z)_{min} > 0\,, \qquad \Delta_3(x,y,z)_{min} > 0\,. \tag{3.61}$$

In contrast, at a *maximizer* $(x,y,z)_{max}$,

$$\Delta_1(x,y,z)_{max} < 0\,, \qquad \Delta_2(x,y,z)_{max} > 0\,, \qquad \Delta_3(x,y,z)_{max} < 0\,. \tag{3.62}$$

Points that are not optimizers, that is, minimizers or maximizers, are *saddle points*.

Exercise 3.19. How would you find and classify the critical points of the following function?

$$f(x,y,z) = \frac{1}{2}\,a_{xx}x^2 + \frac{1}{2}\,a_{yy}y^2 + \frac{1}{2}\,a_{zz}z^2 + a_{xy}xy + a_{xz}xz + a_{yz}yz - a_x x - a_y y - a_z z\,. \tag{3.63}$$

We are now in position to generalize our previous discussion to real-valued function in n variables. For convenience, let us write $\mathbf{x} = (x_1, \ldots, x_n)$. Thus, given a function

$$f\colon U \subseteq \mathbb{R}^n \to \mathbb{R}\,, \qquad \mathbf{x} \mapsto f(\mathbf{x})\,, \tag{3.64}$$

the goal is to optimize it. Depending on the problem, we may be interested in finding the minimizers, $\min f(\mathbf{x})$, or the maximizers, $\max f(\mathbf{x})$. *Critical points*, among them the optimizers (including minimizers and maximizers), are solutions of the following system of equations,

$$\frac{\partial f(\mathbf{x}_c)}{\partial x_1} = 0\,, \quad \frac{\partial f(\mathbf{x}_c)}{\partial x_2} = 0\,, \quad \dots \quad \frac{\partial f(\mathbf{x}_c)}{\partial x_n} = 0\,. \tag{3.65}$$

We can write this system of equations in a more compact form as

$$\nabla f(\mathbf{x}_c) = \mathbf{0}\,, \tag{3.66}$$

where

$$\nabla f(\mathbf{x}_c) = \left(\frac{\partial f(\mathbf{x}_c)}{\partial x_1}, \frac{\partial f(\mathbf{x}_c)}{\partial x_2}, \dots, \frac{\partial f(\mathbf{x}_c)}{\partial x_n} \right). \tag{3.67}$$

Now, to determine the nature of a critical point, that is, to find out whether it is a minimizer, a maximizer, or none of the two, we construct the $n \times n$ *Hessian matrix*,

$$Hf(\mathbf{x}) = \begin{bmatrix} \dfrac{\partial^2 f(\mathbf{x})}{\partial x_1^2} & \cdots & \dfrac{\partial^2 f(\mathbf{x})}{\partial x_1 \partial x_n} \\ \vdots & \ddots & \vdots \\ \dfrac{\partial^2 f(\mathbf{x})}{\partial x_n \partial x_1} & \cdots & \dfrac{\partial^2 f(\mathbf{x})}{\partial x_n^2} \end{bmatrix}. \tag{3.68}$$

We use the Hessian matrix to compute the determinants $\Delta_1(\mathbf{x}_c)$, $\Delta_2(\mathbf{x}_c)$, $\dots$ $\Delta_n(\mathbf{x}_c)$ as in the previous examples (note that these are the determinants of the submatrices starting from the upper left corner of the Hessian matrix). The *second derivative test* states that at a *minimizer* all the determinants are positive, $\Delta_i(\mathbf{x}_{min}) > 0$ for every $i = 1, \dots, n$, whereas at a *maximizer* the determinants alternate sign, $\Delta_1(\mathbf{x}_{max}) < 0$, $\Delta_2(\mathbf{x}_{max}) > 0$, $\Delta_3(\mathbf{x}_{max}) < 0$, and so on. If a point is not an optimizer, then it is a *saddle point*.

Exercise 3.20. Find and classify the critical point of the following function of n variables,

$$f(x_1, \dots, x_n) = \frac{1}{2} \sum_{i=1}^{n} a_{ii} x_i^2 + \sum_{i=1<j}^{n-1} a_{ij} x_i x_j - \sum_{i=1}^{n} a_i x_i\,. \tag{3.69}$$

As we have seen, the statement and steps required to solve an unconstrained continuous optimization problems are, theoretically speaking, rather simple. The real difficulty is to find exact solutions. Because of this, experts usually content themselves with approximate solutions. The most accurate and useful approximate numerical methods they have developed have been adapted to run in modern digital computers. For completeness, let us discuss briefly two of these methods. The first of which was invented long time ago by Newton himself.

The *Newton method* is an iterative process that helps us find approximate values of the critical points of differentiable real-valued functions $f\colon U \subseteq \mathbb{R}^n \to \mathbb{R}$, where n is any positive integer. In other words, the Newton method provides approximate solutions to the system of equations $\nabla f(\mathbf{x}) = \mathbf{0}$. To simplify the presentation, we will write $\nabla f = \mathbf{g}$. So, the critical points of f will correspond to the roots of $\mathbf{g}$.

Because the multivariable case is a straightforward generalization of the Newton method for real-valued functions of a single variable, we first present this easier case. For simplicity, assume that the function $g\colon I \subseteq \mathbb{R} \to \mathbb{R}$, $x \mapsto g(x)$, has only one root x^*, that is, $g(x^*) = 0$. The method starts by choosing two arbitrary points x_0 and x_1 in the neighborhood of the expected value of x^*, and considering the following approximation

$$g'(x_0) = \frac{f(x_1) - f(x_0)}{x_1 - x_0}\,. \tag{3.70}$$

If instead of the actual value of $f(x_1)$, we take $f(x_1) = 0$, we get

$$g'(x_0) = -\frac{f(x_0)}{x_1 - x_0}\,, \tag{3.71}$$

and from here (if $g'(x_0) \neq 0$),

$$x_1 = x_0 - \frac{f(x_0)}{g'(x_0)}\,. \tag{3.72}$$

Exercise 3.21. Draw a picture explaining what we just did.

We now pick another point x_2 and repeat the process, giving

$$x_2 = x_1 - \frac{f(x_1)}{g'(x_1)} = \Big(x_0 - \frac{f(x_0)}{g'(x_0)}\Big) - \frac{f\big(x_0 - f(x_0)/g'(x_0)\big)}{g'\big(x_0 - f(x_0)/g'(x_0)\big)}\,. \tag{3.73}$$

Exercise 3.22. Draw this step.

After k reiterations, we arrive at the point

$$x_k = x_{k-1} - \frac{f(x_{k-1})}{g'(x_{k-1})} = \Big(x_{k-2} - \frac{f(x_{k-2})}{g'(x_{k-2})}\Big) - \frac{f\big(x_{k-2} - f(x_{k-2})/g'(x_{k-2})\big)}{g'\big(x_{k-2} - f(x_{k-2})/g'(x_{k-2})\big)}\,. \tag{3.74}$$

Since every point x_k ($k = 1, 2, \ldots$) has an explicit expression in terms of the precedent point x_{k-1}, any point x_k will ultimately be given by the initial point x_0. The Newton method affirms that the more iterations we perform, the better the approximation value we obtain for x^*. If, as we assumed, $g(x) = f'(x)$, the Newton method can thus be used to find an approximate value of the critical points of f.

Exercise 3.23. Sketched the Newton method described above.

For a multivariable function $\mathbf{g}\colon U \subseteq \mathbb{R}^n \to \mathbb{R}^n$, the procedure is similar. We start by choosing two points $\mathbf{x}_0$ and $\mathbf{x}_1$ in $\mathbb{R}^n$, and applying the total derivative,

$$\mathbf{Dg}(\mathbf{x}_0)(\mathbf{x}_1 - \mathbf{x}_0) = \mathbf{g}(\mathbf{x}_1) - \mathbf{g}(\mathbf{x}_0)\,. \tag{3.75}$$

We then consider $\mathbf{g}(\mathbf{x}_1) = \mathbf{0}$,

$$\mathbf{Dg}(\mathbf{x}_0)(\mathbf{x}_1 - \mathbf{x}_0) = -\mathbf{g}(\mathbf{x}_0)\,. \tag{3.76}$$

If the total derivative is invertible at $\mathbf{x}_0$, then

$$\mathbf{x}_1 = \mathbf{x}_0 - \big(\mathbf{Dg}(\mathbf{x}_0)\big)^{-1}\big(\mathbf{g}(\mathbf{x}_0)\big)\,. \tag{3.77}$$

Exercise 3.24. Try to visualize what is happening here from the geometrical point of view.

In general, after k iterations,

$$\mathbf{Dg}(\mathbf{x}_{k-1})(\mathbf{x}_k - \mathbf{x}_{k-1}) = -\mathbf{g}(\mathbf{x}_{k-1})\,, \tag{3.78}$$

and thus (if the total derivative is invertible at every point $\mathbf{x}_{k-1}$)

$$\mathbf{x}_k = \mathbf{x}_{k-1} - \big(\mathbf{Dg}(\mathbf{x}_{k-1})\big)^{-1}\big(\mathbf{g}(\mathbf{x}_{k-1})\big)\,. \tag{3.79}$$

As for the single variable case, the more iterations, the better the approximate value we obtain for the root $\mathbf{x}^*$ of the function $\mathbf{g}$. Since we are interested specifically in the case $\mathbf{g} = \nabla f$, we conclude that the critical points of f are approximated by

$$\mathbf{x}_k = \mathbf{x}_{k-1} - \big(\mathbf{D}\nabla f(\mathbf{x}_{k-1})\big)^{-1}\big(\nabla f(\mathbf{x}_{k-1})\big)\,. \tag{3.80}$$

There is an additional consideration we have to make which was not necessary in the single variable case. In index notation, the relation $\mathbf{g}(\mathbf{x}) = \nabla f(\mathbf{x})$ takes the form

$$g_j(\mathbf{x}) = \frac{\partial f(\mathbf{x})}{\partial x_j}\,. \tag{3.81}$$

The total derivative $\mathbf{Dg}(\mathbf{x})$, on the other hand, is just the *Jacobian matrix* of $\mathbf{g}(\mathbf{x})$,

$$\mathbf{Dg}(\mathbf{x}) = \left[\frac{\partial g_j(\mathbf{x})}{\partial x_i}\right]. \tag{3.82}$$

Using both relations, we get that

$$\mathbf{Dg}(\mathbf{x}) = \mathbf{D}\nabla f(\mathbf{x}) = \left[\frac{\partial}{\partial x_i}\frac{\partial f(\mathbf{x})}{\partial x_j}\right] = \left[\frac{\partial^2 f(\mathbf{x})}{\partial x_i \partial x_j}\right]. \tag{3.83}$$

This expression is nothing else than the $n \times n$ Hessian matrix (3.68) written in index notation,

$$\mathbf{D}\nabla f(\mathbf{x}) = Hf(\mathbf{x})\,. \tag{3.84}$$

The critical point of f is, therefore, approximately given by

$$\mathbf{x}_k = \mathbf{x}_{k-1} - \big(Hf(\mathbf{x}_{k-1})\big)^{-1}\big(\nabla f(\mathbf{x}_{k-1})\big)\,. \tag{3.85}$$

Note that the Hessian matrix must be invertible.

The second approach we want to mention is the *steepest (or gradient) descent method*. This too is an iterative process. We start by choosing a point $\mathbf{x}_0$, ideally near the expected minimizer we are looking for, and move a distance h, called the *step size*, in the direction opposite to the gradient,

$$\mathbf{x}_1 = \mathbf{x}_0 - h\nabla f(\mathbf{x}_0)\,. \tag{3.86}$$

For the sake of clarity, let us assume for now that the step size h is the same for every iteration of the process. We now use the value $\mathbf{x}_1$ we just found to determine the next best approximation,

$$\mathbf{x}_2 = \mathbf{x}_1 - h\nabla f(\mathbf{x}_1) = \big(\mathbf{x}_0 - h\nabla f(\mathbf{x}_0)\big) - h\nabla f\big(\mathbf{x}_0 - h\nabla f(\mathbf{x}_0)\big)\,. \tag{3.87}$$

After k iterations, the minimizer is approximated by

$$\mathbf{x}_k = \mathbf{x}_{k-1} - h\nabla f(\mathbf{x}_{k-1}) = \big(\mathbf{x}_{k-2} - h\nabla f(\mathbf{x}_{k-2})\big) - h\nabla f\big(\mathbf{x}_{k-2} - h\nabla f(\mathbf{x}_{k-2})\big)\,. \quad (3.88)$$

Since every point $\mathbf{x}_k$ has an explicit expression in terms of the precedent $\mathbf{x}_{k-1}$, any point $\mathbf{x}_k$ will ultimately be given by the initial choice $\mathbf{x}_0$. The iterative process stops when the convergence criteria are met.

Exercise 3.25. Beginning with an arbitrary point $\mathbf{x}_0 = (x_0, y_0)$, write down explicitly the first two steps of the steepest descent method for the function

$$f(x, y) = \frac{1}{2}\, a_{xx}x^2 + \frac{1}{2}\, a_{yy}y^2 + a_{xy}xy\,. \quad (3.89)$$

For simplicity, above we have assumed that the step size is the same for every iteration, however, it is not difficult to convince yourself that this can lead to incorrect results.

Exercise 3.26. Explain visually why this is so?

To remedy this, we can choose a different step size for every iteration. The correct step size is determined as follows. Suppose we define the following function of the step size,

$$s_k(h_k) = f\big(\mathbf{x}_{k-1} - h_k\nabla f(\mathbf{x}_{k-1})\big)\,, \quad (3.90)$$

for $k = 1, 2, \ldots$.

Exercise 3.27. What does this represent? Explain it visually.

The optimal value of h_k is the one that satisfies

$$\begin{aligned} 0 = \frac{ds_k(h_k)}{dh_k} &= \frac{d}{dh_k} f\big(\mathbf{x}_{k-1} - h_k\nabla f(\mathbf{x}_{k-1})\big) \\ &= \nabla f\big(\mathbf{x}_{k-1} - h_k\nabla f(\mathbf{x}_{k-1})\big)\, \frac{d}{dh_k}\big(\mathbf{x}_{k-1} - h_k\nabla f(\mathbf{x}_{k-1})\big) \\ &= \nabla f\big(\mathbf{x}_{k-1} - h_k\nabla f(\mathbf{x}_{k-1})\big)\, \big(-\nabla f(\mathbf{x}_{k-1})\big)\,. \end{aligned} \quad (3.91)$$

Thus, the optimal step size is given by

$$\nabla f\big(\mathbf{x}_{k-1} - h_k\nabla f(\mathbf{x}_{k-1})\big)\, \nabla f(\mathbf{x}_{k-1}) = 0\,. \quad (3.92)$$

Exercise 3.28. Repeat Exercise 3.25 but this time assume that the step sizes are obtained by using the formula (3.92).

3.1.2 Constrained Problems

In contrast to an unconstrained optimization problem, where the set of solutions is completely determined by the domain of the objective function, for *constrained continuous optimization problems* the solution belongs to a smaller set given by the inclusion of additional conditions on the independent variables. The extra conditions are the the so called *constraints*.

In general, given the objective function $f\colon U \subseteq \mathbb{R}^n \to \mathbb{R}$, $\mathbf{x} \mapsto f(\mathbf{x})$, a constrained continuous optimization problem, denoted

$$\min_{\mathbf{x}} f(\mathbf{x}) \quad \text{or} \quad \max_{\mathbf{x}} f(\mathbf{x}) \,, \tag{3.93}$$

(depending whether we want to find the minimizers or the maximizers), can include *m equality constraints*,

$$h_i\colon U_i \subseteq \mathbb{R}^n \to \mathbb{R} \,, \qquad \mathbf{x} \mapsto h_i(\mathbf{x}) = b_i \,, \tag{3.94}$$

with $i = 1, 2, \ldots, m$, and r *inequality constraints*,

$$g_j\colon U_j \subseteq \mathbb{R}^n \to \mathbb{R} \,, \qquad \mathbf{x} \mapsto g_j(\mathbf{x}) \leq d_j \,, \tag{3.95}$$

with $j = 1, 2, \ldots, r$. The constraints can also be written in vector notation as follows,

$$\begin{aligned} &\mathbf{h}\colon U \subseteq \mathbb{R}^n \to \mathbb{R}^m \,, \qquad \mathbf{x} \mapsto \mathbf{h}(\mathbf{x}) = \mathbf{b} \,, \\ &\mathbf{g}\colon U \subseteq \mathbb{R}^n \to \mathbb{R}^r \,, \qquad \mathbf{x} \mapsto \mathbf{g}(\mathbf{x}) \leq \mathbf{d} \,. \end{aligned} \tag{3.96}$$

In a more compact notation, then, a constrained optimization problem is a minimization or maximization problem,

$$\min_{\mathbf{x}} f(\mathbf{x}) \quad \text{or} \quad \max_{\mathbf{x}} f(\mathbf{x}) \,, \tag{3.97}$$

subject to equality and inequality constraints,

$$\mathbf{h}(\mathbf{x}) = \mathbf{b} \,, \qquad \mathbf{g}(\mathbf{x}) \leq \mathbf{d} \,. \tag{3.98}$$

Exercise 3.29. It should be obvious for you that minimization and maximization problems are related by

$$\begin{aligned} \max_{\mathbf{x}} f(\mathbf{x}) &= \min_{\mathbf{x}} \left(-f(\mathbf{x})\right) , \\ \min_{\mathbf{x}} f(\mathbf{x}) &= \max_{\mathbf{x}} \left(-f(\mathbf{x})\right) . \end{aligned} \tag{3.99}$$

Exercise 3.30. How is the classical mechanics' distinction between holonomic and non-holonomic constraints related to our present discussion? Provide at least one mechanical example of each type of constraint.

As we saw above, unconstrained optimization problems are generally difficult to solve. In fact, as soon as the objective function has two or three variables, the solution to the problem can be very difficult to find. Thus, it should not come as a surprise to you to know that, due to the extra restrictions on the decision variables, constrained problems can be even more challenging. Certainly, as the examples discussed in the introduction to this section show, there are constrained problems that are easy to solve. However, these types of problems are an exception. In general, constrained problems are difficult to solve and computer programs are needed. Instead of presenting the most general scenario (something that would take us too far afield from the purpose of these notes), let us simply show how the

Lagrange multipliers method discussed above is generalized to an arbitrary number of decision variables.

Suppose we want to minimize a differentiable function $f\colon U \subseteq \mathbb{R}^n \to \mathbb{R}$, subject to a single equality constraint $h(\mathbf{x}) = 0$. As we already saw (see for example (3.21)), the *Lagrange multipliers method* establishes that at a minimum $\mathbf{x}^*$,

$$\nabla_{\mathbf{x}} f(\mathbf{x}^*) = \mu^* \, \nabla_{\mathbf{x}} h(\mathbf{x}^*) \,. \tag{3.100}$$

The real constant μ^* is the corresponding *Lagrange multiplier* at the minimum. Note that, for the method to work, we must have $\nabla_{\mathbf{x}} h(\mathbf{x}^*) \neq 0$.

If we have several equality constraints, let us say h_i, with $i = 1, 2, \ldots, m$, the generalization is straightforward. We introduce m Lagrange multipliers, one for each constraint, and require that

$$\nabla_{\mathbf{x}} f(\mathbf{x}^*) = \sum_{i=1}^{m} \mu_i^* \, \nabla_{\mathbf{x}} h_i(\mathbf{x}^*) = \boldsymbol{\mu}^* \cdot \nabla_{\mathbf{x}} \mathbf{h}(\mathbf{x}^*) \,, \tag{3.101}$$

where $\boldsymbol{\mu}^* = [\, \mu_1^* \cdots \mu_m^* \,]^T$ is the *Lagrange multipliers vector*. Beware that we have introduced the (unconventional) notation

$$\nabla_{\mathbf{x}} \mathbf{h}(\mathbf{x}^*) = \sum_{i=1}^{m} \nabla_{\mathbf{x}} h_i(\mathbf{x}^*) \, \hat{\mathbf{e}}_i \,, \tag{3.102}$$

where $(\hat{\mathbf{e}}_i)$ is the same ordered basis used to express $\boldsymbol{\mu}^*$,

$$\boldsymbol{\mu}^* = [\, \mu_1^* \cdots \mu_m^* \,]^T = \sum_{i=1}^{m} \mu_i^* \, \hat{\mathbf{e}}_i \,. \tag{3.103}$$

The system of equations (3.101) can conveniently be obtained from a *Lagrangian function*,

$$L(\mathbf{x}, \boldsymbol{\mu}) = f(\mathbf{x}) - \boldsymbol{\mu} \, h(\mathbf{x}) \,, \tag{3.104}$$

and requiring that

$$\nabla_{\mathbf{x}} L(\mathbf{x}^*, \boldsymbol{\mu}^*) = \mathbf{0} \,, \qquad \nabla_{\boldsymbol{\mu}} L(\mathbf{x}^*, \boldsymbol{\mu}^*) = \mathbf{0} \,. \tag{3.105}$$

Since each equality constraint comes with a Lagrange multiplier, this is a system of $n + m$ equations with $n + m$ unknowns.

Exercise 3.31. What if the constraint is $h(\mathbf{x}) = d$ instead of $h(\mathbf{x}) = 0$?

Exercise 3.32. Find the minimizers of

$$f_1(x, y) = x + y \,, \qquad f_2(x, y) = x^2 + y^2 \,, \qquad f_3(x, y) = xy \,, \tag{3.106}$$

subject to each one of the following constraints,

$$h_1(x, y) = x + y = 1 \,, \qquad h_2(x, y) = \frac{x^2}{a^2} + \frac{y^2}{b^2} = 1 \,. \tag{3.107}$$

First, minimize the function by straightforward substitution. Then, compare this result with the one obtained by minimizing using the Lagrange multipliers method. Interpret your results geometrically.

Exercise 3.33. Repeat the previous exercise for

$$f_1(x,y,z) = x + y + z\,, \quad f_2(x,y,z) = x^2 + y^2 + z^2\,, \quad f_3(x,y,z) = xyz\,, \tag{3.108}$$

subject to

$$h_1(x,y,z) = x + y + z = 1\,, \qquad h_2(x,y,z) = \frac{x^2}{a^2} + \frac{y^2}{b^2} + \frac{z^2}{c^2} = 1\,. \tag{3.109}$$

Let us now explain how to solve continuous optimization problems with inequality constraints. Suppose we want to minimize a function $f\colon U \subseteq \mathbb{R}^n \to \mathbb{R}$, subject to a set of r inequality constraints. We compactly write the constraints in vector notation as $\mathbf{g}(\mathbf{x}) \leq \mathbf{d}$. To solve the minimization problem, we construct the Lagrangian function

$$L(\mathbf{x}, \boldsymbol{\lambda}) = f(\mathbf{x}) - \boldsymbol{\lambda} \cdot \big[\, \mathbf{g}(\mathbf{x}) - \mathbf{d} \,\big]\,, \tag{3.110}$$

where $\boldsymbol{\lambda} = [\, \lambda_1 \cdots \lambda_r \,]^T$, and impose,

$$\nabla_{\mathbf{x}} L(\mathbf{x}^*, \boldsymbol{\lambda}^*) = \mathbf{0}\,, \qquad \nabla_{\boldsymbol{\lambda}} L(\mathbf{x}^*, \boldsymbol{\lambda}^*) \leq \mathbf{0}\,, \tag{3.111}$$

with

$$\boldsymbol{\lambda}^* \geq \mathbf{0}\,, \qquad \boldsymbol{\lambda}^* \cdot \big[\, \mathbf{g}(\mathbf{x}^*) - \mathbf{d} \,\big] = 0\,. \tag{3.112}$$

Exercise 3.34. Explain what has to be done to find the minimizers of

$$f(x,y) = a_x x + a_y y\,, \tag{3.113}$$

subject to the inequality constraints

$$g(x,y) = b_x x + b_y y \leq 0\,. \tag{3.114}$$

Exercise 3.35. Repeat the previous exercise for

$$f(x,y) = \frac{1}{2}\, a_{xx} x^2 + \frac{1}{2}\, a_{yy} y^2 + a_{xy} xy - a_x x - a_y y\,, \tag{3.115}$$

subject to

$$g_1(x,y) = \frac{1}{2}\, b_{xx} x^2 + \frac{1}{2}\, b_{yy} y^2 + b_{xy} xy - b_x x - b_y y \leq 0\,, \tag{3.116}$$

$$g_2(x,y) = \frac{1}{2}\, c_{xx} x^2 + \frac{1}{2}\, c_{yy} y^2 + c_{xy} xy - c_x x - c_y y \leq 0\,. \tag{3.117}$$

Choose some values of the constants so that you can explicitly solve the problem.

The generalization to minimization problems with equality as well as inequality constraints, $\min f(\mathbf{x})$ subject to $\mathbf{h}(\mathbf{x}) = \mathbf{b}$ and $\mathbf{g}(\mathbf{x}) \leq \mathbf{d}$, is straightforward. We define a Lagrangian function

$$L(\mathbf{x}, \boldsymbol{\lambda}, \boldsymbol{\mu}) = f(\mathbf{x}) - \boldsymbol{\mu} \cdot \big[\, \mathbf{h}(\mathbf{x}) - \mathbf{b} \,\big] - \boldsymbol{\lambda} \cdot \big[\, \mathbf{g}(\mathbf{x}) - \mathbf{d} \,\big]\,. \tag{3.118}$$

The solution of the minimization problem uses a combination of the two methods discussed above. Because of the equality constraints we require that,

$$\nabla_{\mathbf{x}} L(\mathbf{x}^*, \boldsymbol{\lambda}^*, \boldsymbol{\mu}^*) = \mathbf{0}\,, \qquad \nabla_{\boldsymbol{\mu}} L(\mathbf{x}^*, \boldsymbol{\lambda}^*, \boldsymbol{\mu}^*) = \mathbf{0}\,. \tag{3.119}$$

On the other hand, the inequality constraints impose that,

$$\nabla_{\mathbf{x}} L(\mathbf{x}^*, \boldsymbol{\lambda}^*, \boldsymbol{\mu}^*) = \mathbf{0}\,, \qquad \nabla_{\boldsymbol{\lambda}} L(\mathbf{x}^*, \boldsymbol{\lambda}^*, \boldsymbol{\mu}^*) \leq \mathbf{0}\,, \tag{3.120}$$

with

$$\boldsymbol{\lambda}^* \geq \mathbf{0}\,, \qquad \boldsymbol{\lambda}^* \cdot \big[\, \mathbf{g}(\mathbf{x}^*) - \mathbf{d} \,\big] = 0\,. \tag{3.121}$$

If unconstrained optimization problems are generally difficult to solve, just imagine how hard are problems with mixed constraints. This is the reason why iterative processes such as the Newton method and the steepest descent method, and their advanced versions, are used to approximate the solutions. You can look at the literature if you are interested in these sorts of problems. In the remaining of this subsection, we will discuss some important special cases. Let us start with the classic *diet problem*.

Suppose that there are P food products available in the market, individually denoted by $1, 2, \ldots, p, \ldots, P$, and we know their prices, $c_1, c_2, \ldots c_p, \ldots, c_P$. If every day we buy a quantity x_p of product p, the total daily cost will then be,

$$C(x_1, x_2, \ldots, x_P) = c_1 x_1 + c_2 x_2 + \ldots + c_P x_P = \sum_{p=1}^{P} c_p x_p\,. \tag{3.122}$$

Let us say we are interested in keeping track of the consumption of N nutrients and we know that product p contains a quantity n_{ip} of nutrient i. The daily consumption of nutrient i is then,

$$D_i = n_{i1} x_1 + n_{i2} x_2 + \ldots + n_{iP} x_P\,. \tag{3.123}$$

A similar relation holds for all the other nutrients, $i = 1, 2, \ldots, N$. Now, suppose nutritionists recommend that every person should consume at least a quantity R_i of nutrient i. This requirement imposes that,

$$D_i = n_{i1} x_1 + n_{i2} x_2 + \ldots + n_{iP} x_P \geq R_i\,, \tag{3.124}$$

for every $i = 1, 2, \ldots, N$. The diet problem aims at finding the minimum daily expenses given the recommendations of the nutritionists. In other words, the idea is to find the right combination and amount of each food product so that the nutrient requirements are satisfied with the least possible cost. In mathematical language,

$$\min_{(x_1, \ldots, x_P)} \sum_{p=1}^{P} c_p x_p\,, \tag{3.125}$$

subject to

$$\begin{aligned} n_{11} x_1 + n_{12} x_2 + \ldots + n_{1P} x_P &\geq R_1\,, \\ \vdots \qquad\quad & \vdots \qquad\quad \vdots \\ n_{N1} x_1 + n_{N2} x_2 + \ldots + n_{NP} x_P &\geq R_N\,. \end{aligned} \tag{3.126}$$

This optimization problem can be written more compactly in terms of vectors and matrices. For example, we can collect all the prices c_p and quantities x_p in appropriate column vectors $[c_p] = \mathbf{c}$ and $[x_p] = \mathbf{x}$, so that

$$\min_{(x_1, \ldots, x_P)} \sum_{p=1}^{P} c_p x_p = \min_{\mathbf{x}} \mathbf{c}^T \mathbf{x}\,. \tag{3.127}$$

Similarly, we can group all the coefficients n_{ip} in an $N \times P$ matrix $N = [n_{ip}]$, so that the nutrient requirements can be simply written as

$$N\mathbf{x} \geq \mathbf{R}\,. \tag{3.128}$$

Exercise 3.36. Some nutrients not only have a minimum required quantity, but also a suggested maximum consumption quantity. How would you formulate a problem with these sorts of constraints?

Exercise 3.37. Suppose some nutrients have to be consumed in an exact quantity, how would you reformulate the diet problem?

The diet problem is an example of a *linear optimization problem* or *linear program*, for short. They are the object of study of a subfield of optimization theory known as *linear programming*. A linear program is an optimization problem with a *linear objective function*,

$$\min_{\mathbf{x}} \mathbf{a}^T\mathbf{x} \quad \text{or} \quad \max_{\mathbf{x}} \mathbf{a}^T\mathbf{x}\,, \tag{3.129}$$

subject to *linear constraints*,

$$H\mathbf{x} = \mathbf{b}\,, \qquad G\mathbf{x} \leq \mathbf{d}\,. \tag{3.130}$$

Note that, if there are m equality constraints, then $\mathbf{b}$ is an $m \times 1$ column vector and, assuming there are n unknowns, H is an $m \times n$ matrix. Similarly, if there are r inequality constraints, then $\mathbf{d}$ is an $r \times 1$ column vector and G is a $r \times n$ matrix.

Be cautious that there are several equivalent ways of defining a linear program. For instance, the inequality constraint is sometimes written $G\mathbf{x} \geq \mathbf{d}$. These two definitions are certainly equivalent because we can simply multiply both sides of the inequality constraint by -1 and define the matrix $G' = -G$ and the constant vector $\mathbf{d}' = -\mathbf{d}$, giving $G'\mathbf{x} \leq \mathbf{d}'$. It is said that a linear minimization problem is in *standard form* if it is written as follows,

$$\min_{\mathbf{x}} \mathbf{a}^T\mathbf{x}\,, \quad \text{subject to} \quad H\mathbf{x} = \mathbf{b}\,, \quad \mathbf{x} \geq \mathbf{0}\,. \tag{3.131}$$

Here, the non-negativity condition on the decision variables is explicitly indicated.

Exercise 3.38. Show that any linear program can be written in standard form.

Exercise 3.39. State mathematically two real-life linear programs.

Optimization problems that are not linear, either because the objective function or the constraints are not linear, are called *nonlinear optimization problems*. They are the object of study of *nonlinear programming* and, in general, they are extremely difficult to solve (even with computer programs). Among these, we are particularly interested in optimization problems with *quadratic objective functions*,

$$f(x_1, \ldots, x_n) = \frac{1}{2}\sum_{i,j=1}^{n} a_{ij}x_i x_j + \sum_{i=1}^{n} a_i x_i\,, \tag{3.132}$$

where all the coefficients are constant and $a_{ij} = a_{ji}$. We can use column vectors and matrices to write them in a more compact form. We first rewrite

$$f(x_1, \ldots, x_n) = \frac{1}{2}\sum_{i,j=1}^{n} x_i a_{ij} x_j + \sum_{i=1}^{n} a_i x_i\,. \tag{3.133}$$

We then define the symmetric matrix

$$A = \begin{bmatrix} a_{11} & \dots & a_{1n} \\ \vdots & \ddots & \vdots \\ a_{n1} & \dots & a_{nn} \end{bmatrix} . \tag{3.134}$$

Using the column vectors $[x_i] = \mathbf{x}$, $[a_i] = \mathbf{a}$ and the matrix $A = [a_{ij}]$, the quadratic function above takes the following form,

$$f(\mathbf{x}) = \frac{1}{2}\,\mathbf{x}^T A\mathbf{x} + \mathbf{a}^T\mathbf{x}\,. \tag{3.135}$$

Quadratic optimization problems, also known as *quadratic programs*, are optimization problems with quadratic objective functions and linear constraints. For example, a quadratic minimization problem,

$$\min_{\mathbf{x}} \frac{1}{2}\,\mathbf{x}^T A\mathbf{x} + \mathbf{a}^T\mathbf{x}\,, \tag{3.136}$$

is subject to linear constraints,

$$H\mathbf{x} = \mathbf{b}\,, \qquad G\mathbf{x} \leq \mathbf{d}\,. \tag{3.137}$$

Exercise 3.40. Use column vectors and matrices to write all the objective functions and constraints in Exercises 3.32 to 3.35. Classify the programs as linear, quadratic or none of the two.

3.2 Dual Optimization Problems

So far we have avoided discussing functions with multiple extreme points. However, we know that most functions have several, if not infinite optimizers. For example, the potential $V = cx^2(x^2 - x_0^2)$ has more than one optimizer. These sorts of problems are very common in physics, however, they are generally difficult to solve. That is why we are particularly interested in problems with a single optimizer. Convex optimization deals with problems with convex objective functions, that have this property.

Exercise 3.41. Find the optimizers of the function $V = cx^2(x^2 - x_0^2)$. Draw it and explain under which conditions could the Newton method fail to solve it.

There have been many progresses in the theory of convex optimization. Most notably, the fact that convex optimization are solvable in polynomial time. This, among many other suitable properties, has replacement of the obsolete division between linear and nonlinear problems with the convex vs nonconvex optimization problems. For example, linear and quadratic functions are convex, so their corresponding problems fall under the umbrella of convex optimization. After some basic definitions of duality theory, we will move to duality.

Let M be a metric space, that is, a set of points with a metric (distance) function, and X be an open subset of M. We say that X is a *convex set* of M if given any two points $x, y \in X$, the points

$$z = x + t(y - x) = ty + (t - 1)x\,, \tag{3.138}$$

for all values of the parameter $t \in [0, 1]$, are in X.

Exercise 3.42. Draw two and three-dimensional examples of convex and non- convex sets.

Consider, for example, the two-dimensional case, which is easy to visualize. Let $U \subseteq \mathbb{R}^2$ and $\mathbf{x}$ and $\mathbf{y}$ two points in U. The subset U of $\mathbb{R}^2$ is convex if the points

$$\mathbf{z} = t\mathbf{y} + (t-1)\mathbf{x}\,, \tag{3.139}$$

for $t \in [0,1]$, are in U. Now, let $f\colon I \subseteq \mathbb{R} \to \mathbb{R}$, $x \mapsto f(x) = y$. The area above the graph of f is called the *epigraph* of f,

$$\operatorname{epi} f = \{(x, Y) \in I \times \mathbb{R} \mid Y \geq y\}\,. \tag{3.140}$$

When the epigraph is a convex set, we say that f is a *convex function*. In higher dimensions, the definitions are similar. The function $f\colon U \subseteq \mathbb{R}^n \to \mathbb{R}$, $\mathbf{x} \mapsto f(x)$, is a *convex function* if the epigraph of f is a convex set of $\mathbb{R}^{n+1}$.

Exercise 3.43. Provide all the details of the above definitions.

Exercise 3.44. Show that linear and quadratic functions are convex.

A *convex optimization problem*, or *convex program*, consists in the optimization of a convex function $f\colon U \subseteq \mathbb{R}^n \to \mathbb{R}$, $\mathbf{x} \mapsto f(x)$,

$$\min_{\mathbf{x}} f(\mathbf{x}) \quad \text{or} \quad \max_{\mathbf{x}} f(\mathbf{x})\,, \tag{3.141}$$

subject to linear equality constraints,

$$H\mathbf{x} = \mathbf{b}\,, \tag{3.142}$$

and convex inequality constraints,

$$\mathbf{g}(\mathbf{x}) \leq \mathbf{d}\,. \tag{3.143}$$

Note that, while the equality constraints must be linear functions, the inequality constraints are convex functions (they can also, of course, be linear).

As we said, convex programming is a vast field and we will not discuss it further here. In addition to the fact that they have only one minimizer and they are solvable in polynomial time, we are interested in one additional feature of convex problems: the so called *(Lagrange) duality*. This property allows us to reformulate a convex problem, the *primal program*, as another which is easier to solve, the *dual program*. The general proof of this theorem is beyond the purpose of these notes. The idea, though, is simple: solve a minimization problem in the decision variable $\mathbf{x}$ by solving the maximization problem in the dual variable $\boldsymbol{\mu}$ and $\boldsymbol{\lambda}$ (the Lagrange multipliers). Let us see a couple of examples to illustrate how the method works.

Suppose we want to solve the linear program,

$$\max_{\mathbf{x}} \mathbf{a}^T\mathbf{x}\,, \quad \text{subject to} \quad H\mathbf{x} = \mathbf{b} \quad \text{and} \quad \mathbf{x} \geq \mathbf{0}\,. \tag{3.144}$$

The Lagrange multipliers method tells us to construct the Lagrangian function,

$$L(\mathbf{x}, \boldsymbol{\mu}) = \mathbf{a}^T\mathbf{x} + \boldsymbol{\mu}^T(H\mathbf{x} - \mathbf{b})\,, \tag{3.145}$$

and the solutions will be given by the system of equations

$$\nabla_{\mathbf{x}} L(\mathbf{x}^*, \boldsymbol{\mu}^*) = 0\,, \qquad \nabla_{\boldsymbol{\mu}} L(\mathbf{x}^*, \boldsymbol{\mu}^*) = 0\,. \tag{3.146}$$

We thus extend the space of independent variables from n to $n+m$ (assuming that they are m equality constraints) and solve the problem as if it was an unconstrained problem. But, suppose we rewrite the Lagrangian function as follows

$$\begin{aligned} L(\mathbf{x}, \boldsymbol{\mu}) &= -\boldsymbol{\mu}^T \mathbf{b} + (\mathbf{a}^T + \boldsymbol{\mu}^T H)\,\mathbf{x} \\ &= -\mathbf{b}^T \boldsymbol{\mu} + \mathbf{x}^T (\mathbf{a} + H^T \boldsymbol{\mu}) \\ &= -\big(\mathbf{b}^T \boldsymbol{\mu} - \mathbf{x}^T (H^T \boldsymbol{\mu} - \mathbf{a})\big)\,. \end{aligned} \tag{3.147}$$

Rewritten in this form, the prescription (3.146) gives the solution to the problem

$$\min_{\boldsymbol{\mu}} \mathbf{b}^T \boldsymbol{\mu}\,, \quad \text{subject to} \quad H^T \boldsymbol{\mu} \geq \mathbf{a}\,. \tag{3.148}$$

This is the dual program of (3.144).

Exercise 3.45. Show that the linear program,

$$\max_{\mathbf{x}} \mathbf{a}^T \mathbf{x}\,, \quad \text{subject to} \quad G\mathbf{x} \leq \mathbf{d} \quad \text{and} \quad \mathbf{x} \geq \mathbf{0}\,, \tag{3.149}$$

has as dual program

$$\min_{\boldsymbol{\lambda}} \mathbf{d}^T \boldsymbol{\lambda}\,, \quad \text{subject to} \quad G^T \boldsymbol{\lambda} = \mathbf{a} \quad \text{and} \quad \boldsymbol{\lambda} \geq \mathbf{0}\,. \tag{3.150}$$

3.3 Integer Programs

Until now, we have only discussed optimization problems with continuous decision variables. However, many real-life optimization problems involve discontinuous variables. For example, in the case of the diet problem, there are food products that are not sold in fractional parts; you can only buy them in an integer multiple of a fixed quantity. Optimization problems with nothing but integer variables are called *integer optimization problems* or *integer programs*. Some of them include constraints, the so called *constrained integer programs*, and some do not, the *unconstrained integer programs*.

The applications of linear programs to science and engineering are countless. The standard example is the so called *knapsack problem*. Suppose you have a knapsack with a limiting weight capacity W_{max} and a set of N items you want to take with you. The goal is to maximize the total value of the content of the knapsack,

$$\max_{(n_1,\ldots,n_N)} c_1 n_1 + \ldots + c_i n_i + \ldots + c_N n_N = \max_{\mathbf{n}} \mathbf{c}^T \mathbf{n}\,, \tag{3.151}$$

where c_i is the cost of the ith item and its quantity $n_i \in \mathbb{Z}^{\geq 0}$. The weight of the stuff you put in the knapsack is

$$W = w_1 n_1 + \ldots + w_N n_N = \mathbf{w}^T \mathbf{n}\,. \tag{3.152}$$

Since the total weight cannot exceed W_{max}, we have to impose the constraint

$$\mathbf{w}^T \mathbf{n} \leq W_{max}\,. \tag{3.153}$$

Thus, the knapsack problem consists in

$$\max_{\mathbf{n}} \mathbf{c}^T\mathbf{n}\,, \quad \text{subject to} \quad \mathbf{w}^T\mathbf{n} \leq W_{max} \quad \text{and} \quad \mathbf{n} \in (\mathbb{Z}^{\geq 0})^N\,. \tag{3.154}$$

A general *linear integer program* consists in

$$\min_{\mathbf{z}} \mathbf{a}^T\mathbf{z} \quad \text{or} \quad \max_{\mathbf{z}} \mathbf{a}^T\mathbf{z}\,, \tag{3.155}$$

subject to

$$H\mathbf{z} = \mathbf{b}\,, \quad G\mathbf{z} \leq \mathbf{d} \quad \text{and} \quad \mathbf{z} \in (\mathbb{Z}^{\geq 0})^N\,. \tag{3.156}$$

Exercise 3.46. Think about some real-life integer linear programs.

Consider again the diet problem, but this time assume that some of the items are sold exclusively in integer numbers. In this case, the problem involves continuous as well as integer variables. We denote by c, with $c = 1, \ldots, C$, a product that can be bought in continuous quantities and by i, with $i = 1, \ldots, I$, a product that can be bought only in integer numbers. Their respective prices are denoted by c_c and c_i. The objective function is then

$$\begin{aligned} C(\mathbf{x}, \mathbf{y}) &= (c_1x_1 + \ldots + c_cx_c + \ldots + c_Cx_C) + (c_1z_1 + \ldots + c_iz_i + \ldots + c_Iz_I) \\ &= \mathbf{c}_C^T\,\mathbf{x} + \mathbf{c}_I^T\,\mathbf{z}\,. \end{aligned} \tag{3.157}$$

Here we are using the column vector notation $\mathbf{x} = [\,x_1 \ldots x_C\,]^T \in (\mathbb{R}^{\geq 0})^C$ and $\mathbf{z} = [\,z_1 \ldots z_I\,]^T \in (\mathbb{Z}^{\geq 0})^I$. The constraint, as we already said, is given by the minimum daily requirement of each nutrient. Since each nutrient can be included in both continuous and discrete food products (see (3.128)), the inequality constraints are compactly written as

$$N_C\mathbf{x} + N_I\mathbf{z} \geq \mathbf{R}\,. \tag{3.158}$$

This is an example of a *mixed integer linear program*. In general, they can involve continuous as well integer decision variables and the constraints, both equality and inequality constraints, are linear. Note that a mixed integer linear program with $\mathbf{z} = \mathbf{0}$ is purely continuous. Similarly, when $\mathbf{x} = \mathbf{0}$, the problem is purely integer.

Exercise 3.47. Think about some real-life mixed integer linear programs.

Solutions by approximating to integer numbers. For example, in elementary optimization problems when the solution obtained involves fractional people, objects or other things that cannot be divided.

Of special interest to us are optimization problems where the decision variables can only take the values zero and one. These problems are called binary optimization problems. Of course, we can have *unconstrained binary optimization problems* and *constrained binary optimization problems*. In this section we will briefly discuss some of these problems, in particular unconstrained binary optimization problems (for reasons we will see in the next section). For completeness, let us formally state them.

A *binary program* is an optimization problem with objective function defined on binary variables. That is, given $\mathbf{b} = [\,b_1 \ldots b_n\,]$, where $b_i \in \{0, 1\}$ for $i = 1, 2, \ldots, n$, the objective function is of the type

$$\mathbf{f}\colon \{0,1\}^n \to \mathbb{R}\,, \qquad \mathbf{b} \mapsto \mathbf{f}(\mathbf{b})\,. \tag{3.159}$$

Since the decision variables are binary, the constraint functions (in case they are) must also be defined on binary variables.

Consider again the knapsack problem. This time, however, suppose that the knapsack can at most contain one item of each type. The *binary knapsack problem* is stated exactly as above, but now the decision variables are zeros and ones. In real-life terms, the item is included or not. It then consists in

$$\max_{\mathbf{b}} \mathbf{c}^T\mathbf{b}\,, \quad \text{subject to} \quad \mathbf{w}^T\mathbf{b} \leq W_{max} \quad \text{and} \quad \mathbf{b} \in \{0,1\}^n\,. \tag{3.160}$$

In general, a *binary linear program* is an integer linear program with binary, rather than integer variables. That is, a binary linear program consists in optimizing a binary linear function subject to binary linear equality and inequality constraints.

To conclude, consider a *binary quadratic function*,

$$f(\mathbf{b}) = \alpha\,\mathbf{b}^T Q\,\mathbf{b} + \beta\,\mathbf{c}^T\mathbf{b}\,, \tag{3.161}$$

where $\mathbf{b} \in \{0,1\}^n$, Q is a symmetric $n \times n$ matrix and α and β are real numbers. In index notation,

$$f(b_1 \dots b_n) = \alpha \sum_{i,j=1}^{n} b_i Q_{ij} b_j + \beta \sum_{i=1}^{n} c_i b_i\,, \tag{3.162}$$

where $b_i \in \{0,1\}$ and $Q_{ij} = Q_{ji}$. A *quadratic unconstrained binary optimization problem (QUBO problem)* aims to find the

$$\min_{\mathbf{b}} \alpha\,\mathbf{b}^T Q\,\mathbf{b} + \beta\,\mathbf{c}^T\mathbf{b} \quad \text{or} \quad \max_{\mathbf{b}} \alpha\,\mathbf{b}^T Q\,\mathbf{b} + \beta\,\mathbf{c}^T\mathbf{b}\,, \tag{3.163}$$

where $\mathbf{b} \in \{0,1\}^n$ is the only restriction on the decision variables $b_1, \dots, b_n$.

Exercise 3.48. Suppose that the decision variables do not take the values 0 and 1 but -1 and 1. In this case, how would you state a binary problem?

Exercise 3.49. Show that any QUBO problem can be written as,

$$\min_{\mathbf{b}} \gamma\,\mathbf{b}^T Q'\,\mathbf{b}\,, \tag{3.164}$$

where Q' is a symmetric matrix and γ is a real constant.

4 Classical Portfolio Optimization Theory

In this section, we discuss the so called *modern portfolio theory*. This is a standard topic in financial mathematics and the literature on the subject is vast. Here, we only review the main concepts and results.

Suppose you have a certain amount of money, cash for that matter, and you want to invest it in the stock market with the expectation, of course, of making a profit over a given period of time. The problem you face is the following: what is the best choice of stocks and the optimal amount of money you have to invest in each of them so that, at due time, you receive the maximum return on your investment? This is, in simple words, the *portfolio optimization problem*. The summary below is rather complete and self-contained for the purposes of this review.

In Subsection 4.1, we explain the basics of modern portfolio theory and in Subsection 4.2, we show how this mathematical model is related to continuous optimization. The case of discrete portfolios is discussed in Subsection 4.3. In particular, we discuss QUBO problems, the sorts of problems future quantum computers are expected to tackle efficiently.

4.1 Mathematical Description of an Investment Portfolio

In our modern globalized world, the domestic and international markets offer a wide variety of options to invest your cash: real estate, government bonds, foreign currencies, precious metals, cryptocurrencies, and many more. When you exchange your cash for any of these *assets*, your intention is to make a future financial benefit. The way this will benefit you can vary and can be difficult to track. For example, some investments will save you money and others will produce you a cash inflow. What is important, though, is that the purpose of owning any of these assets, whatever their nature, is to increase your wealth.

Out of all these possibilities, we will only consider *stocks (shares)* of companies traded in the public market. Since the price of a single stock represents a small percentage of the value of the entire company, by acquiring a stock, you own a small part of it. Why we are particularly interested in this financial asset will become clear in the following pages. For now, let it suffice to say that we want to consider assets whose prices change rapidly and unpredictably.

Suppose you invest all your money in S stocks, $s = 1, 2, \ldots, S$. This is the *price of your portfolio* at the initial time $t = 0$ and is denoted by $p_P(0)$. Assume, moreover, that the initial *price of the stock* s is $p_s(0)$. Since the initial price of the portfolio is the sum of the initial prices of the individual stocks, we have that

$$p_P(0) = \sum_{s=1}^{S} p_s(0)\,. \tag{4.1}$$

The *stock weight* of stock s is defined as the ratio

$$w_s(0) = \frac{p_s(0)}{p_P(0)}\,. \tag{4.2}$$

Using these relations,

$$p_P(0) = \sum_{s=1}^{S} p_s(0) = \sum_{s=1}^{S} w_s(0) p_P(0)$$

$$= p_P(0) \sum_{s=1}^{S} w_s(0)\,, \tag{4.3}$$

giving that (as expected)

$$\sum_{s=1}^{S} w_s(0) = 1\,. \tag{4.4}$$

Formally speaking, an *(investment) portfolio* is a given set of initial stock weights,

$$\{w_1(0), w_2(0), \ldots, w_S(0)\} = \{w_s(0)\}_{s=1}^{S} = \{w_s(0)\}\,. \tag{4.5}$$

Different sets correspond to different portfolios.

If we want to know whether an initial investment has produced a profit or a loss over a given period of time, we have to, somehow, compare the initial and final values of the portfolio. To simplify the following presentation, we first consider the case of a single stock and then generalize to a multi-stock portfolio.

Given the values of a stock at two different times, $p_s(0)$ and $p_s(T)$, the simplest way to compare them is simply by subtracting them. For every stock s, we define the *stock return* from $t = 0$ to $t = T$ as,

$$r_s(T) = p_s(T) - p_s(0)\,. \tag{4.6}$$

When $r_s(T) > 0$, we say that the initial investment on the stock s produced a *profit*. In simple words, you made money. If, on the contrary, $r_s(T) < 0$, then we say that the initial investment produced a *loss*. That is, you lost money. Finally, when $r_s(T) = 0$, the investment on the stock s did not produce any profit or loss and your final wealth is exactly the same as the one you had at the beginning of the transaction. This is, without a doubt, an extremely simple mathematical model of a real situation that usually involves many other factors. For example, on the positive side, a more realistic model could include dividends and interests on the profit. On the other hand, on the negative side, you may want to include service fees, taxes and inflation rates. This simple model, though, is the one we will discuss here. More realistic scenarios can be found in the specialized literature.

Similar to the stock return, the *portfolio return* from $t = 0$ to $t = T$ is given by the difference of the portfolio prices,

$$r_P(T) = p_P(T) - p_P(0)\,. \tag{4.7}$$

When $r_P(T) > 0$, the portfolio produces a profit, and when $r_P(T) < 0$, it produces a loss. It is easy to show that this equation is equivalent to the sum of all the stock returns over the same period of time,

$$r_P(T) = \sum_{s=1}^{S} r_s(T)\,. \tag{4.8}$$

Exercise 4.1. Prove the previous equation.

Now that we know how much profit (or loss) does a portfolio produce, we want to compare the performance of different portfolios. In order to compare two portfolios P_1 and P_2, we could, for instance, subtract their returns,

$$\Delta r_{P_1 P_2}(T) = r_{P_1}(T) - r_{P_2}(T)\,. \tag{4.9}$$

The choice between the two portfolios seems obvious: just pick the portfolio with the highest return. There is a practical problem with this, though. The problem is that two portfolios with different prices at the beginning of the transaction can produce equal returns. From the investor's point of view it is, of course, more convenient the portfolio that requires the least initial investment. For this reason, we introduce the concept of rate of return.

The *return rate* of the stock s from $t = 0$ to $t = T$ is defined as

$$R_s(T) = \frac{r_s(T)}{p_s(0)} = \frac{p_s(T) - p_s(0)}{p_s(0)}\,. \tag{4.10}$$

Since $p_s(0)$ is a positive quantity, the interpretation of the sign of $R_s(T)$ is the same as the one given above for $r_s(T)$.

The *portfolio return rate* from $t = 0$ to $t = T$ is

$$\begin{aligned} R_P(T) = \frac{r_P(T)}{p_P(0)} &= \frac{\sum_{s=1}^{S} r_s(T)}{p_P(0)} \\ &= \sum_{s=1}^{S} \frac{p_s(0)}{p_P(0)} R_s(T) = \sum_{s=1}^{S} w_s(0) R_s(T)\,. \end{aligned} \tag{4.11}$$

Since $p_P(0)$ is a positive quantity, the interpretation of the sign of $R_P(T)$ is the same as the one given above for $r_P(T)$.

Let us use some basic linear algebra to rewrite all the previous formulas in a simpler way. *Price vectors* are collections of stock prices,

$$\mathbf{p}(0) = \begin{bmatrix} p_1(0) \\ \vdots \\ p_S(0) \end{bmatrix}\,, \qquad \mathbf{p}(T) = \begin{bmatrix} p_1(T) \\ \vdots \\ p_S(T) \end{bmatrix}\,. \tag{4.12}$$

An investment portfolio is specified by its *weight vector* at the initial time,

$$\mathbf{w}(0) = \begin{bmatrix} w_1(0) \\ \vdots \\ w_S(0) \end{bmatrix}\,. \tag{4.13}$$

Condition (4.4) becomes

$$\mathbf{1}^T \mathbf{w}(0) = 1\,, \tag{4.14}$$

where the vector $\mathbf{1} = [\,1 \ldots 1\,]^T$. Stock return rates are collected in the *(portfolio) return rate vector*,

$$\mathbf{R}(T) = \begin{bmatrix} R_1(T) \\ \vdots \\ R_S(T) \end{bmatrix}\,. \tag{4.15}$$

Using this compact notation, the portfolio return rate (4.11) is written

$$R_P(T) = \mathbf{w}^T(0)\mathbf{R}(T)\,. \tag{4.16}$$

So far we have only considered two moments in time, $t = 0$ and $t = T$. Suppose now two periods of time: $[\,0, T\,]$ and $[\,T, 2T\,]$. Correspondingly, we have the price vectors $\mathbf{p}(0)$, $\mathbf{p}(T)$ and $\mathbf{p}(2T)$, the weight vectors $\mathbf{w}(0)$ and $\mathbf{w}(T)$, and the return rate vectors $\mathbf{R}(T)$ and $\mathbf{R}(2T)$. The portfolio return rates during these two periods are given by

$$R_P(0,T) = \mathbf{w}^T(0)\mathbf{R}(T)\,, \qquad R_P(T,2T) = \mathbf{w}^T(T)\mathbf{R}(2T)\,. \tag{4.17}$$

Note that we have introduced a more explicit notation to denote the portfolio return rates for each of the periods. It is clear that,

$$R_P(0,2T) = R_P(0,T) + R_P(T,2T)\,. \tag{4.18}$$

We define the *average portfolio return rate* as

$$\mu_P(0,2T) = \frac{1}{2}\,R_P(0,2T) = \frac{1}{2}\,\big(R_P(0,T) + R_P(T,2T)\big)\,. \tag{4.19}$$

To know whether in a period of time we have, on average, gain or lose money, we compute

$$R_P(0,T) - \mu_P(0,T)\,, \qquad R_P(T,2T) - \mu_P(0,2T)\,. \tag{4.20}$$

If the quantity if positive (negative), we are making more (less) money than the average. Of course, since $R_P(0,T) = \mu_P(0,T)$, this quantity is only relevant for more than one periods.

We can generalize all this to N time periods, $[\,0, T\,]$, $[\,T, 2T\,]$, $\ldots$, $[\,(N-1)T, NT\,]$. In this case, the portfolio return rate at the end of the nth period, for $n = 1, 2, \ldots N$, is

$$\begin{aligned} R_P(0,nT) &= R_P(0,T) + R_P(T,2T) + \ldots + R_P\big((n-1)T, nT\big) \\ &= \sum_{k=1}^{n} R_P\big((k-1)T, kT\big)\,, \end{aligned} \tag{4.21}$$

with corresponding average portfolio return,

$$\mu_P(0,nT) = \frac{1}{n}\,R_P(0,nT) = \frac{1}{n}\,\sum_{k=1}^{n} R_P\big((k-1)T, kT\big)\,, \tag{4.22}$$

and average profit (or loss) in the nth period given by

$$R_P\big((n-1)T, nT\big) - \mu_P(0,nT)\,. \tag{4.23}$$

For simplicity, in the following we will only assume one period of time ($N = 1$). The stocks are bought at time $t = 0$ and the return, whether a profit or a loss, is decided at time $T > 0$. What happens in between is unknown to us. This is the *single-period investment model*.

4.2 The Mean-Variance Model

Suppose again that you own a small part of a big company because you bought one or more of the company's stocks. Your expectation as stockholder, that is, as owner of a small percentage of the company, is to benefit from an increase in the company's worth. However, due to unknown and unpredictable factors, the future value of the company is not certain. It may increase, as you expect, but it may also decrease, in which case you lose money. There are so many factors at play that even the prediction of the short-term future of the company's worth, and consequently the return on your investment, seems impossible.

The belief that future prices are unpredictable goes back to the pioneering observations made by Louis Bachelier at the beginning of the 20th century. More than one century of historical records have confirmed (though, fair to say, not in a conclusive manner) that the evolution of the stock prices do not follow any recognizable pattern. In other words, in finance it is assumed that the analysis of the historical price changes and economic factors cannot be exploited to predict the future of stock prices. In fact, the movement of the stock prices is described by a random-walk (the same phenomenon studied in many areas of physics).

The *random-walk theory of stock prices* assumes that every market player has immediate access to all the market information available. So, in case there is a short-term outperformance of the market by a group of traders, other investors will immediately recognize the trend and their actions will have an opposite effect on the market, ultimately restoring randomness.

Note: Since stock prices are random, in the following Box we summarize the main probability concepts necessary to understand prices and investment portfolios.

Box 4.1. Probability of discrete random variables.

Suppose you perform an experiment many, many times and you always obtain the same finite number of results, say $\omega_1, \omega_2, \ldots, \omega_n$. We collect all these outcomes and form the so called *sample space*,

$$\Omega = \{\omega_1, \omega_2, \ldots, \omega_n\} = \{\omega_i\}_{i=1}^n \,. \tag{4.24}$$

A *discrete random variable* is a function X on Ω that assigns a real value to every possible outcome ω_i,

$$X : \Omega \to \mathbb{R} \,, \qquad \omega_i \mapsto X(\omega_i) = x_i \,. \tag{4.25}$$

If n_i is the number of occurrences of the outcome ω_i and N the number of experiments (that we assume is as large as necessary), we assign to the outcome ω_i a *probability of occurrence* $p_{\omega_i} = p_i = n_i/N$. The set of all pairs (ω_i, p_{ω_i}),

$$\mathcal{P} = \{(\omega_i, p_{\omega_i})\}_{i=1}^n \,, \tag{4.26}$$

is known as the *probability distribution* (more precisely, since the number of outcomes is countable, this is a *probability mass function*). From the probabilistic point of view, the probability distribution completely characterizes our system.

More abstractly, we define a *probability function* as a relation that assigns to every value x_i the probability of occurrence of outcome ω_i,

$$P\colon \{x_i\}_{i=1}^n \to [0,1]\,, \qquad x_i = X(\omega_i) \mapsto P\big(X(\omega_i)\big) = P(x_i) = p_i\,. \tag{4.27}$$

The notation $P(X = x_i)$ to denote p_i is also of common use. This probability function, by definition, has the following properties. Firstly, the probability that an outcome ω_i is in Ω is one, usually written $P(\Omega) = 1$. Secondly, the probability that the experiment gives no result in Ω is zero, written $P(\varnothing) = 0$. Thirdly and finally,

$$P(A) = \sum_{\alpha} P\big(X(\omega_\alpha)\big)\,, \tag{4.28}$$

where $A \subseteq \Omega$ and $\omega_\alpha \in A$. The set of outcomes A is called an *event*. In particular, as we pointed out,

$$P(\Omega) = \sum_{i=1}^{n} P\big(X(\omega_i)\big) = 1\,. \tag{4.29}$$

The expectation value of a random variable is a quantity that relies on the historical data provided by the probability distribution. The *expected value* of a discrete random variable is also known as its *average* or *mean*, and is given by

$$\mathbb{E}(X) = \mu_X = \frac{1}{N}\sum_{i=1}^{n} n_i\, x_i = \sum_{i=1}^{n} \frac{n_i}{N}\, x_i = \sum_{i=1}^{n} p_i\, x_i\,. \tag{4.30}$$

The *variance* measures the variability of the historical values of a random variable. It is defined as follows,

$$\begin{aligned} \mathrm{Var}(X) = \sigma_X^2 &= \frac{1}{N}\sum_{i=1}^{n} n_i\,(x_i - \mu_X)^2 \\ &= \sum_{i=1}^{n} \frac{n_i}{N}\,(x_i - \mu_X)^2 = \sum_{i=1}^{n} p_i\,(x_i - \mu_X)^2 \\ &= \mathbb{E}[(X - \mu_X)^2]\,. \end{aligned} \tag{4.31}$$

The variance measures the spread of the values a random variable with respect to the mean. The greater the variance, the greater its variability and unpredictability. Instead of using the variance, we can estimate the spread of a set of data by the *standard deviation*,

$$\sigma_X = \sqrt{\mathrm{Var}(X)} = \sqrt{\mathbb{E}[(X - \mu_X)^2]}\,. \tag{4.32}$$

The advantage of the standard deviation over the variance is simply that σ_X has the same units as x_i. This makes it easier to visualize both quantities in the same plot.

The common behavior of two random variables X_1 and X_2 can be measured by using a new quantity known as the *covariance* between X_1 and X_2,

$$\sigma_{12} = \mathrm{Cov}(X_1, X_2) = \mathbb{E}[(X_1 - \mu_1)(X_2 - \mu_2)]\,. \tag{4.33}$$

In index notation,

$$\sigma_{12} = \sum_{i,j=1}^{n} p\big(x_{1,i}, x_{2,j}\big)\,(x_{1,i} - \mu_1)(x_{2,j} - \mu_2)$$
$$= \sum_{i,j=1}^{n} p_{i,j}\,(x_{1,i} - \mu_1)(x_{2,j} - \mu_2)\,, \tag{4.34}$$

where $p_{i,j} = p\big(x_{1,i}, x_{2,j}\big)$ is the *joint probability*, that is, the probability that both $x_{1,i}$ and $x_{2,j}$ occur at the same time. Using that $p_{i,j} = p_{j,i}$ and renaming the indices, we conclude that

$$\sigma_{21} = \sum_{i,j=1}^{n} p\big(x_{2,i}, x_{1,j}\big)\,(x_{2,i} - \mu_2)(x_{1,j} - \mu_1)$$
$$= \sum_{i,j=1}^{n} p\big(x_{1,i}, x_{2,j}\big)\,(x_{1,i} - \mu_1)(x_{2,j} - \mu_2) = \sigma_{12}\,. \tag{4.35}$$

Note that, variance and covariance are related by

$$\sigma_{11} = \mathrm{Cov}(X_1, X_1) = \mathbb{E}[(X_1 - \mu_1)^2] = \sigma_1^2\,. \tag{4.36}$$

Given two random variables, it is convenient to collect the variances and covariances in a symmetric 2×2 *covariance matrix*,

$$\Sigma(X_1, X_2) = \begin{bmatrix} \sigma_{11} & \sigma_{12} \\ \sigma_{21} & \sigma_{22} \end{bmatrix} = \begin{bmatrix} \sigma_1^2 & \sigma_{12} \\ \sigma_{12} & \sigma_2^2 \end{bmatrix}. \tag{4.37}$$

An $S \times S$ covariance matrix can as well be defined for multiple random variables X_1, X_2, …, X_S,

$$\Sigma\big(X_1, X_2, \ldots, X_S\big) = \begin{bmatrix} \sigma_{11} & \ldots & \sigma_{1S} \\ \vdots & \ldots & \vdots \\ \sigma_{S1} & \ldots & \sigma_{SS} \end{bmatrix} = \begin{bmatrix} \sigma_1^2 & \ldots & \sigma_{1S} \\ \vdots & \ldots & \vdots \\ \sigma_{1S} & \ldots & \sigma_S^2 \end{bmatrix}. \tag{4.38}$$

Since, according to the accepted hypothesis, stock prices behave as random variables, we can only make probabilistic predictions about future stock prices. More specifically, given the historical data of a stock price, we can calculate the expected value (arithmetic mean of past prices) and the variance (uncertainty of future prices). Suppose the sample space $\Omega = \{\omega_i\}_{i=1}^n$ contains all the possible market scenarios at time $T > 0$. We define the *price of the stock* s at time T as the random variable,

$$p_s(T)\colon \Omega \to \mathbb{R}^{\geq 0}\,, \qquad \omega_i \mapsto \big(p_s(T)\big)(\omega_i) = p_s(T, \omega_i)\,. \tag{4.39}$$

Note that the codomain of the random variable $p_s(T)$ has been constrained to $\mathbb{R}^{\geq 0}$ because prices cannot be negative. As for any other random variable, definition

(4.30) provides the expected price of the stock s at time T,

$$\mathbb{E}\big(p_s(T)\big) = \sum_{i=1}^{n} P\big(p_s(T,\omega_i)\big)\, p_s(T,\omega_i) = \sum_{i=1}^{n} p_i\, p_s(T,\omega_i)\,. \tag{4.40}$$

Recall that the return of the stock s during the time period $[0,T]$ is given by the formula (4.6),

$$r_s(T) = p_s(T) - p_s(0)\,. \tag{4.41}$$

Now, since the price of the stock at time T, $p_s(T)$, is a random variable, it follows that the *return of the stock* at time T, $r_s(T)$, is also a random variable,

$$r_s(T)\colon \Omega \to \mathbb{R}\,, \qquad \omega_i \mapsto \big(r_s(T)\big)(\omega_i) = r_s(T,\omega_i)\,. \tag{4.42}$$

Note that the codomain of the random variable $r_s(T)$ is the entire real line because returns can be negative (loss), zero or positive (profit). The *return rate of stock* s, given by (4.10),

$$R_s(T) = \frac{r_s(T)}{p_s(0)} = \frac{p_s(T) - p_s(0)}{p_s(0)}\,, \tag{4.43}$$

is, obviously, also a random variable,

$$R_s(T)\colon \Omega \to \mathbb{R}\,, \qquad \omega_i \mapsto \big(R_s(T)\big)(\omega_i) = R_s(T,\omega_i)\,. \tag{4.44}$$

The *expected return rate* of the stock s is then,

$$\mathbb{E}\big(R_s(T)\big) = \mathbb{E}\left(\frac{p_s(T) - p_s(0)}{p_s(0)}\right). \tag{4.45}$$

We can write it in terms of the expected price of the stock s at time T as follows,

$$\mathbb{E}\big(R_s(T)\big) = \frac{\mathbb{E}\big(p_s(T)\big)}{p_s^2(0)}\,. \tag{4.46}$$

Exercise 4.2. Prove this relation.

If the expected return rate is positive (negative), we anticipate a profit (loss) from our investment. To have an estimation of the variability of the price of the stock, and ultimately an estimation of the uncertainty of our investment, we use the *variance of the return rate*,

$$\mathrm{Var}\big(R_s(T)\big) = \sum_{i=1}^{n} p_i \left[R_s(T,\omega_i) - \mathbb{E}\big(R_s(T)\big)\right]^2. \tag{4.47}$$

To clarify the ideas, consider the simple case of two possible market scenarios ω_1 and ω_2. We assume that we know the initial price of the stock, $p_s(0)$, and the probabilities p_1 and p_2 of the two scenarios to occur. Moreover, suppose we know that, if ω_1 (ω_2) occurs, the price of the stock becomes $p_s(T,\omega_1)$ ($p_s(T,\omega_2)$). The expected price of the stock at time T is then,

$$\mathbb{E}\big(p_s(T)\big) = p_1 p_s(T,\omega_1) + p_2 p_s(T,\omega_2)\,. \tag{4.48}$$

We can use this result to find the expected return rate,

$$\mathbb{E}\big(R_s(T)\big) = \frac{p_1 p_s(T,\omega_1) + p_2 p_s(T,\omega_2)}{p_s(0)^2}\,. \tag{4.49}$$

The variance, on the other hand, is given by

$$\mathrm{Var}\big(R_s(T)\big) = p_1\big[R_s(T,\omega_1) - \mathbb{E}\big(R_s(T)\big)\big] + p_2\big[R_s(T,\omega_2) - \mathbb{E}\big(R_s(T)\big)\big]\,. \tag{4.50}$$

Exercise 4.3. Do it for three possible scenarios.

For actual financial portfolios, we have to consider multiple stocks, the prices of which are random variables. Consider, for example, a portfolio with two stocks s and s'. Suppose an initial investment,

$$p_P(0) = w_s(0)p_P(0) + w_{s'}(0)p_P(0)\,. \tag{4.51}$$

As we saw in (4.11), the portfolio return rate at a later time T is given by

$$R_P(T) = w_s(0)R_s(T) + w_{s'}(0)R_{s'}(T)\,. \tag{4.52}$$

Assume now two possible future scenarios ω_1 and ω_2, with corresponding probabilities of occurrence p_1 and p_2. Moreover, suppose we know the prices of the stocks in each case: $p_s(T,\omega_1)$, $p_{s'}(T,\omega_1)$, $p_s(T,\omega_2)$, and $p_{s'}(T,\omega_2)$. The *expected portfolio return rate* is

$$\mathbb{E}\big(R_P(T)\big) = w_s(0)\mathbb{E}\big(R_s(T)\big) + w_{s'}(0)\mathbb{E}\big(R_{s'}(T)\big)\,, \tag{4.53}$$

where

$$\mathbb{E}\big(R_s(T)\big) = p_1 R_s(T,\omega_1) + p_2 R_s(T,\omega_2)\,, \tag{4.54}$$

$$\mathbb{E}\big(R_{s'}(T)\big) = p_1 R_{s'}(T,\omega_1) + p_2 R_{s'}(T,\omega_2)\,. \tag{4.55}$$

The *variance of the portfolio return rate* is

$$\mathrm{Var}\big(R_P(T)\big) = p_1\big[R_P(T,\omega_1) - \mathbb{E}\big(R_P(T)\big)\big]^2 + p_2\big[R_P(T,\omega_2) - \mathbb{E}\big(R_P(T)\big)\big]^2\,. \tag{4.56}$$

Exercise 4.4. Show that this equation is equivalent to saying that

$$\begin{aligned}
\mathrm{Var}\big(R_P(T)\big) &= w_s^2(0)\mathrm{Var}\big(R_s(T)\big) + w_{s'}^2(0)\mathrm{Var}\big(R_{s'}(T)\big) \\
&\quad + 2p_1 w_s(0)w_{s'}(0)\big[R_s(T,\omega_1) - \mathbb{E}\big(R_s(T)\big)\big]\big[R_{s'}(T,\omega_1) - \mathbb{E}\big(R_{s'}(T)\big)\big] \\
&\quad + 2p_2 w_s(0)w_{s'}(0)\big[R_s(T,\omega_2) - \mathbb{E}\big(R_s(T)\big)\big]\big[R_{s'}(T,\omega_2) - \mathbb{E}\big(R_{s'}(T)\big)\big] \\
&= w_s^2(0)\mathrm{Var}\big(R_s(T)\big) + w_{s'}^2(0)\mathrm{Var}\big(R_{s'}(T)\big) \\
&\quad + 2w_s(0)w_{s'}(0)\sum_{i=1}^{2} p_i\big[R_s(T,\omega_i) - \mathbb{E}\big(R_s(T)\big)\big]\big[R_{s'}(T,\omega_i) - \mathbb{E}\big(R_{s'}(T)\big)\big]\,.
\end{aligned}$$

Using the covariance between two random variables (4.33), we arrive at

$$\begin{aligned}
\mathrm{Var}\big(R_P(T)\big) &= w_s^2(0)\mathrm{Var}\big(R_s(T)\big) + w_{s'}^2(0)\mathrm{Var}\big(R_{s'}(T)\big) \\
&\quad + 2w_s(0)w_{s'}(0)\mathrm{Cov}\big(R_s(T), R_{s'}(T)\big)\,.
\end{aligned} \tag{4.57}$$

For simplicity, we switch to the sigma notation for variances and covariances,

$$\mathrm{Var}\big(R_P(T)\big) = \sigma_P^2(T)\,, \tag{4.58}$$

$$\mathrm{Var}\big(R_s(T)\big) = \sigma_{ss}(T) = \sigma_s^2(T)\,, \tag{4.59}$$

$$\mathrm{Cov}\big(R_s(T), R_{s'}(T)\big) = \sigma_{ss'}(T)\,. \tag{4.60}$$

With this notation, the variance of the portfolio return rate becomes

$$\sigma_P^2(T) = w_s^2(0)\sigma_s^2(T) + w_{s'}^2(0)\sigma_{s'}^2(T) + 2w_s(0)w_{s'}(0)\sigma_{ss'}(T)\,. \tag{4.61}$$

Exercise 4.5. Suppose a portfolio composed of two stocks s and s' that satisfy

$$\sigma_{ss'}(T) = \sigma_s(T)\sigma_{s'}(T)\,, \tag{4.62}$$

where $\sigma_s(T), \sigma_{s'}(T) \neq 0$. Show that the variance of the portfolio return rate is zero when the initial investment is such that

$$w_s(0) = \frac{\sigma_{s'}(T)}{\sigma_{s'}(T) - \sigma_s(T)}\,. \tag{4.63}$$

How much is invested in the stock s'? What happens when $\sigma_s(T) > \sigma_{s'}(T)$? Prove that the lower the expected return rate of the stock s', the worthier the investment in this portfolio.

Exercise 4.6. Suppose a portfolio of two uncorrelated stocks s and s' and assume that we know the expected return rates and variances of the two individual stocks. Prove that the variance of the portfolio return rate is

$$\sigma_P^2(T) = \left(\frac{\mathbb{E}\big(R_P(T)\big) - R_{s'}(T)}{\mathbb{E}\big(R_s(T)\big) - R_{s'}(T)}\right)^2 \sigma_s^2(T) + \left(\frac{\mathbb{E}\big(R_s(T)\big) - \mathbb{E}\big(R_P(T)\big)}{\mathbb{E}\big(R_s(T)\big) - R_{s'}(T)}\right)^2 \sigma_{s'}^2(T)\,.$$

What is the expected portfolio return rate corresponding to a minimum variance? Show that the portfolio is given by

$$w_s(0) = \frac{\sigma_{s'}^2}{\sigma_s^2 + \sigma_{s'}^2}\,, \qquad w_{s'}(0) = \frac{\sigma_s^2}{\sigma_s^2 + \sigma_{s'}^2}\,. \tag{4.64}$$

The generalization to portfolios with more than two stocks, $s = 1, 2, \ldots, S$, is straightforward. The expected portfolio return rate is simply,

$$\mathbb{E}\big(R_P(T)\big) = \sum_{s=1}^{S} w_s(0)\,\mathbb{E}\big(R_s(T)\big)\,, \tag{4.65}$$

where, assuming n possible future scenarios,

$$\mathbb{E}\big(R_s(T)\big) = \sum_{i=1}^{n} p_i R_s(T, \omega_i)\,. \tag{4.66}$$

A slightly shorter notation writes

$$\mathbb{E}\big(R_P(T)\big) = \mu_P(T)\,, \qquad \mathbb{E}\big(R_s(T)\big) = \mu_s(T)\,, \tag{4.67}$$

yielding

$$\mu_P(T) = \sum_{s=1}^{S} w_s(0)\,\mu_s(T)\,. \tag{4.68}$$

The *weight vector* $\mathbf{w}(0) = [\,w_1(0)\ \ldots\ w_S(0)\,]^T$ and the *vector of expected return rates* $\boldsymbol{\mu}(T) = [\,\mu_1(T)\ \ldots\ \mu_S(T)\,]^T$ can be used to give

$$\mu_P(T) = \mathbf{w}^T(0)\boldsymbol{\mu}(T) = \boldsymbol{\mu}^T(T)\mathbf{w}(0)\,. \tag{4.69}$$

Similarly, the variance of the return rate of a portfolio of S stocks is given by summing over all stocks $s, s' = 1, 2, \dots S$ in (4.61),

$$\begin{aligned}\sigma_P^2(T) &= \sum_{s=1}^{S} w_s^2(0)\sigma_s^2(T) + 2\sum_{\substack{s,s'=1\\ s\neq s'}}^{S} w_s(0)w_{s'}(0)\sigma_{ss'}(T)\\ &= \sum_{s=1}^{S} w_s(0)\sigma_{ss}(T)w_s(0)\\ &+ \sum_{\substack{s,s'=1\\ s<s'}}^{S} w_s(0)\sigma_{ss'}(T)w_{s'}(0) + \sum_{\substack{s,s'=1\\ s>s'}}^{S} w_s(0)\sigma_{ss'}^2(T)w_{s'}(0)\,. \end{aligned} \tag{4.70}$$

We use the general covariance matrix (4.38) corresponding to S random variables,

$$\Sigma\big(R_1(T), \dots, R_S(T)\big) = \Sigma\big(\mathbf{R}(T)\big) = \begin{bmatrix} \sigma_{11}(T) & \dots & \sigma_{1S}(T) \\ \vdots & \dots & \vdots \\ \sigma_{S1}(T) & \dots & \sigma_{SS}(T) \end{bmatrix}, \tag{4.71}$$

and the weight vector of S stocks, $\mathbf{w}(0) = [\, w_1(0) \ \dots \ w_S(0)\,]^T$, to write the *variance of the portfolio return rate* in a more compact form,

$$\sigma_P^2(T) = \mathbf{w}^T(0)\,\Sigma\big(\mathbf{R}(T)\big)\,\mathbf{w}(0)\,. \tag{4.72}$$

From here, we obtain the standard deviation,

$$\sigma_P(T) = \sqrt{\mathbf{w}^T(0)\,\Sigma\big(\mathbf{R}(T)\big)\,\mathbf{w}(0)}\,. \tag{4.73}$$

Associated to the variance of a portfolio return rate is the concept of *risk*. Risk is, simply put, the possibility of losing money from an investment. The greater the *volatility* of the individual stocks in a portfolio, that is, the standard deviation of their prices, the greater the possibility of losing money. Since two or more stocks in a portfolio can be correlated, it is indeed the covariance matrix that measures the risk of an investment portfolio. Of course, not all assets are risky. Think about, for example, the cash you keep at home or the money in the bank earning a fixed interest. These sorts of assets are called *risk-free* or *riskless assets*. Here, we are not interested in them, though. We are only interested in *risky assets*, for instance, stocks whose prices change randomly in time.

In the *modern portfolio theory*, also known as the *mean-variance model*, it is assumed that risk is directly proportional to the variance of the portfolio return rate. That is, the higher the variance of the portfolio return rate, the higher the risk. Note that, according to this definition, risk also implies the possibility of making more money than expected. However, risk usually has a negative connotation and it is interpreted as the possibility of making less money than expected or to lose money. If we define *risk tolerance*, denoted by $\mathcal{R}$, as the level of risk an investor is willing to take, we have that the portfolio $\mathbf{w}(0)$ must be constructed such that

$$\mathbf{w}^T(0)\,\Sigma\big(\mathbf{R}(T)\big)\,\mathbf{w}(0) \leq \mathcal{R}\,. \tag{4.74}$$

Equivalently, we can write

$$\alpha\,\mathbf{w}^T(0)\,\Sigma\big(\mathbf{R}(T)\big)\,\mathbf{w}(0) = \mathcal{R}\,, \tag{4.75}$$

where the positive quantity α is the *risk aversion coefficient*.

Exercise 4.7. Use this definition of risk to interpret the Exercises 4.5 and 4.6.

Before moving to the optimization of portfolios, something that we will discuss in the next subsection, let us examine some additional facts about the modern portfolio theory. For simplicity, let us consider portfolios with only two stocks.

Suppose that the variance of the return rate of each stock is known, σ_s^2 and $\sigma_{s'}^2$, as well as the covariance of their return rates, $\sigma_{ss'}$. Recalling that $w_s(0) + w_{s'}(0) = 1$, we can write the standard deviation of the portfolio return rate (4.61) exclusively in terms of $w_s(0)$,

$$\sigma_P(T) = \sqrt{w_s^2(0)\sigma_s^2(T) + \big(1 - w_s(0)\big)^2\sigma_{s'}^2(T) + 2w_s(0)\big(1 - w_s(0)\big)\sigma_{ss'}(T)}\,. \tag{4.76}$$

Let us try to visualize this. First, we define the function

$$\gamma_1 \colon [0,1] \to \mathbb{R}\,, \qquad w_s(0) \mapsto \gamma_1\big(w_s(0)\big) = \sigma_P(T)\,. \tag{4.77}$$

This function on the single variable $w_s(0)$ substitutes $\sigma_P(T)$,

$$\gamma_1\big(w(0)\big) = \sqrt{w_s^2(0)\sigma_s^2(T) + \big(1 - w_s(0)\big)^2\sigma_{s'}^2(T) + 2w_s(0)\big(1 - w_s(0)\big)\sigma_{ss'}(T)}\,. \tag{4.78}$$

Assuming that we know the expected return rate of each stock, $\mu_s(T)$ and $\mu_{s'}(T)$, and using $w_{s'}(0) = 1 - w_s(0)$, the expected portfolio return rate (4.68) becomes

$$\mu_P(T) = w_s(0)\mu_s(T) + \big(1 - w_s(0)\big)\mu_{s'}(T)\,. \tag{4.79}$$

We now define a second function,

$$\gamma_2 \colon [0,1] \to \mathbb{R}\,, \qquad w_s(0) \mapsto \gamma_2\big(w_s(0)\big) = \mu_P(T)\,. \tag{4.80}$$

That is, we substitute $\mu_P(T)$ by the function γ_2 of $w_s(0)$,

$$\gamma_2\big(w_s(0)\big) = w_s(0)\mu_s(T) + \big(1 - w_s(0)\big)\mu_{s'}(T)\,. \tag{4.81}$$

We then use these functions to define a parameterized curve on the Cartesian plane,

$$\boldsymbol{\gamma} = \begin{bmatrix}\gamma_1\\ \gamma_2\end{bmatrix} : [0,1] \to \mathbb{R}^2\,, \quad w_s(0) \mapsto \boldsymbol{\gamma}\big(w_s(0)\big) = \begin{bmatrix}\gamma_1\big(w_s(0)\big)\\ \gamma_2\big(w_s(0)\big)\end{bmatrix} = \begin{bmatrix}\sigma_P(T)\\ \mu_P(T)\end{bmatrix}\,. \tag{4.82}$$

The image $\{\boldsymbol{\gamma}[0,1]\}$ is called the *risk curve*.

Exercise 4.8. Show that the risk curve is a hyperbola.

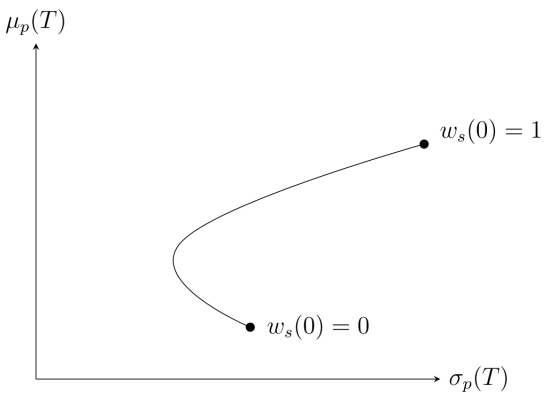

Fig. 2. The risk curve of a two-stock portfolio.

The risk curve helps visualize at a glance how the expected portfolio return rate is related to the investment risk. The horizontal axis measures the risk, $\sigma_P(T)$, and the vertical axis measures the expected return rate, $\mu_P(T)$. An investor can, for example, decide in advance an acceptable level of risk and then look at the risk curve to check how much return is expected for such level of risk. Alternatively, the investor can decide in advance how much return he or she expects from the investment and then consider if the risk is tolerable. The risk curve confirms the common knowledge that the higher (lesser) the risk, the higher (lesser) the potential profit or loss.

Note that, for some risk levels, the risk curve has two possible expected portfolio return rates. Since investors are interested in portfolios with the highest expected return rate, we can discard the lower part of the curve. That is, instead of $w_s(0) \in [0,1]$, we have $w_s(0) \in [a,1]$, with a the value of $w_s(0)$ where the risk curve has a vertical slope. We can, therefore, substitute the two-dimensional parameterized curve $\boldsymbol{\gamma}$ by the real-value function

$$\bar{\mu}_P \colon [\gamma_1(a), \gamma_1(1)] \to \mathbb{R}\,, \qquad \sigma_P(T) \mapsto \bar{\mu}_P\big(\sigma_P(T)\big)\,. \tag{4.83}$$

Exercise 4.9. Find the explicit expression of $\bar{\mu}_P\big(\sigma_P(T)\big)$.

The graph of the function $\bar{\mu}_P$,

$$G(\bar{\mu}_P) = \big\{\big(\sigma_P(T), \bar{\mu}_P\big(\sigma_P(T)\big)\big) \mid \sigma_P(T) \in \big[\gamma_1(a), \gamma_1(1)\big]\big\}\,, \tag{4.84}$$

is known as the *efficient frontier*. Points in the efficient frontier correspond to *optimal* or *efficient portfolios*. Since $\bar{\mu}_P$ is a monotonically increasing function, portfolios with higher (lesser) potential return rates are uniquely associated with higher (lesser) risks

If points in the efficient frontier are associated to efficient portfolios, that is, portfolios that maximize expected return rates for given levels of risk, points below the efficient frontier correspond to *suboptimal* or *inefficient portfolios*. An inefficient portfolio is one for which the values of $w_s(0)$ and $w_{s'}(0)$ are such that the investment risk is not worth the potential return. In fact, all the points to the left of an inefficient portfolio have the same expected return rate but with lesser investment risk. In mathematical language, an inefficient portfolio is associated to a point

$$\big(\sigma_P(T), \bar{\mu}_P\big(\sigma_P(T)\big)\big) \in \operatorname{hyp}_{\mathrm{s}} \bar{\mu}_P\,, \tag{4.85}$$

where the *strict hypograph* of $\bar{\mu}_P$ is the set,

$$\operatorname{hyp}_{\mathrm{s}} \bar{\mu}_P = \big\{(\sigma_P(T), Y) \in \big[\gamma_1(a), \gamma_1(1)\big] \times \mathbb{R} \mid Y < \bar{\mu}_P\big(\sigma_P(T)\big)\big\}\,. \tag{4.86}$$

The *attainable set* are all the points in the Cartesian plane associated to efficient and inefficient portfolios. Mathematically speaking, these are the points in the graph and the strict hypograph of $\bar{\mu}_P$. In other words, the attainable set contains all the points in the *hypograph* of $\bar{\mu}_P$,

$$\operatorname{hyp} \bar{\mu}_P = \big\{(\sigma_P(T), Y) \in \big[\gamma_1(a), \gamma_1(1)\big] \times \mathbb{R} \mid Y \leq \bar{\mu}_P\big(\sigma_P(T)\big)\big\}\,. \tag{4.87}$$

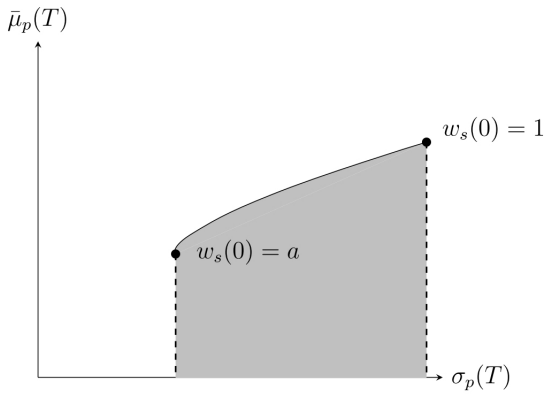

Fig. 3. The efficient frontier and the attainable set.

To conclude, let us see how the shape of a risk curve depends on the covariance of the expected return rates of the stocks s and s'. We define the *correlation coefficient* as

$$\rho_{ss'}(T) = \frac{\sigma_{ss'}(T)}{\sigma_s(T)\sigma_{s'}(T)}\,. \tag{4.88}$$

Because

$$-\sigma_s(T)\sigma_{s'}(T) \le \sigma_{ss'}(T) \le \sigma_s(T)\sigma_{s'}(T)\,, \tag{4.89}$$

it follows that

$$-1 \le \rho_{ss'}(T) \le +1\,. \tag{4.90}$$

Exercise 4.10. Prove (4.89).

The full expression of $\rho_{ss'}(T)$ is

$$\rho_{ss'}(T) = \frac{\sum_{ij} p\big(R_{s,i}(T), R_{s',j}(T)\big)\big[R_{s,i}(T) - \mu_s(T)\big]\big[R_{s,j}(T) - \mu_{s'}(T)\big]}{\sqrt{\sum_i p(R_{s,i}(T))\big[R_{s,i}(T) - \mu_s(T)\big]^2}\sqrt{\sum_j R_{s',j}(T))\big[R_{s',j}(T) - \mu_{s'}(T)\big]^2}}\,, \tag{4.91}$$

where we have written $R_{s,i}(T) = R_s(T, \omega_i)$ and the coefficients $p\big(R_{s,i}(T), R_{s',j}(T)\big)$ are the joint probabilities (see (4.34) for details) .

Since the covariance of two random variables measures the degree of relation between them, so does the correlation coefficient. In our case, the correlation coefficient measures the relation between the expected return rates of the two stocks s and s'. If $\rho_{ss'}(T) = -1$, we say that the expected return rates of the two stocks are "perfectly negative correlated". What this means is that when the expected return rate of one of the stocks is the highest, the expected return rate of the second stock is the lowest. When $\rho_{ss'}(T) = +1$, we say that the expected return rates of the two stocks are "perfectly positive correlated". That is, if the expected return rate of one stock is the highest (lowest), the same happens with the other. Finally, when $\rho_{ss'}(T) = 0$, the return rates are uncorrelated. Two stocks are said to be uncorrelated when the return rate of one of them tells us nothing about the other. Between perfectly negative an perfectly positive correlation, there is a wide variety of situations. Roughly speaking, when the correlation coefficient is near -1 or $+1$, we say that the prices are strongly negative or positive correlated. On the other hand, when it is near 0, we say that they are weakly correlated. The exact use of these degrees of correlation depends on the specific situation under study.

Using the correlation coefficient $\rho_{ss'}(T)$, the standard deviation of the portfolio return rate becomes

$$\sigma_P(T) = \sqrt{w_s^2(0)\sigma_s^2(T) + \big(1 - w_s(0)\big)^2\sigma_{s'}^2(T) + 2w_s(0)\big(1 - w_s(0)\big)\rho_{ss'}(T)\sigma_s(T)\sigma_{s'}(T)}\,. \tag{4.92}$$

Exercise 4.11. Use the result of Exercise (4.9) to write $\bar{\mu}_P\big(\sigma_P(T)\big)$ in terms of the correlation coefficient $\rho_{ss'}(T)$. Plot $\bar{\mu}_P\big(\rho_{ss'}(T)\big)$ for several values of $\rho_{ss'}(T)$ between -1 and $+1$.

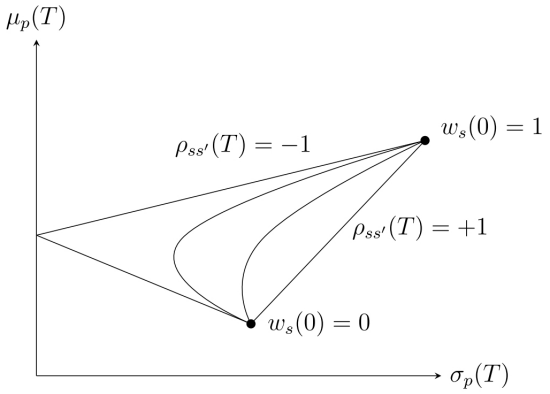

Fig. 4. Risk curves for correlation coefficients between -1 and $+1$.

In order to mitigate risk, a savvy investor will try to choose stocks that are as negative correlated as possible. The same strategy applies to portfolios with more than two stocks. Ideally, the portfolio is constructed with stocks that are perfectly negative correlated. That is, if the portfolio contains S stocks, any two stocks s and s' in S should ideally satisfy $-1 \lesssim \rho_{ss'}$. By proceeding in this way, the investor assures that the losses of some stocks are balanced by the gains generated by the other stocks. This risk management strategy is what is know as *diversification*. A diversified portfolio comes, however, with a cost: it decreases the chances of losing money, but at the same time it reduces the potential reward.

4.3 Portfolio Optimization as a Quadratic Programming

An investor will try to maximize the expected return rate of his or her portfolio while at the same he or she tries to keep the risk at a tolerable level. The choice of the portfolio, that is, of the weight vector $\mathbf{w}(0)$, determines the portfolio return rate according to the formula (4.69),

$$\mu_P(T) = \boldsymbol{\mu}^T(T)\mathbf{w}(0)\,, \tag{4.93}$$

where $\mathbf{1}^T\mathbf{w}(0) = 1$. On the other hand, the risk is proportional to the variance of the portfolio return rate (4.75),

$$\mathcal{R} = \alpha\,\sigma_P^2(T) = \alpha\,\mathbf{w}^T(0)\,\Sigma\big(\mathbf{R}(T)\big)\,\mathbf{w}(0)\,. \tag{4.94}$$

The optimization problem the investor has to solve is,

$$\max_{\mathbf{w}(0)}\ \boldsymbol{\mu}^T(T)\mathbf{w}(0)\,, \tag{4.95}$$

subject to

$$\alpha\, \mathbf{w}^T(0)\, \Sigma\big(\mathbf{R}(T)\big)\, \mathbf{w}(0) = \mathcal{R}\,, \tag{4.96}$$

and $\mathbf{1}^T\mathbf{w}(0) = 1$. In the following, we will simplify the notation by simply writing,

$$\max_{\mathbf{w}} \boldsymbol{\mu}^T\mathbf{w}\,, \quad \text{subject to} \quad \alpha\, \mathbf{w}^T\, \Sigma\, \mathbf{w} = \mathcal{R} \quad \text{and} \quad \mathbf{1}^T\mathbf{w} = 1\,. \tag{4.97}$$

Since an optimal portfolio has a unique expected return for every level of risk, and vice versa, instead of choosing a level of risk and look for the maximum return, the investor can equivalently choose an expected return $\mathcal{M}$ and look for the minimum risk. The optimization problem thus becomes,

$$\min_{\mathbf{w}} \alpha\, \mathbf{w}^T\, \Sigma\, \mathbf{w}\,, \quad \text{subject to} \quad \boldsymbol{\mu}^T\mathbf{w} = \mathcal{M} \quad \text{and} \quad \mathbf{1}^T\mathbf{w} = 1\,. \tag{4.98}$$

So far we have considered portfolios with continuous decision variables $w_s(0) \in (0, 1)$, for every $s = 1, 2, \ldots, S$. In order to convert this problem into a binary problem, we use the following mathematical trick. First, recall that the stock weight of a stock $s = 1, 2, \ldots, S$, is given by (4.2),

$$w_s(0) = \frac{p_s(0)}{p_P(0)}\,. \tag{4.99}$$

Suppose we now introduce a quantity,

$$\widetilde{w}_s(0) = \begin{cases} 0 & \text{if the stock } s \text{ is not in the portfolio,} \\ w_s(0) & \text{if the stock } s \text{ is in the portfolio.} \end{cases}$$

It is easy to check that nothing changes in the previous pages if we substitute $w_s(0)$ by $\widetilde{w}_s(0)$. Equivalently, we can write $\widetilde{w}_s(0)$ as

$$\widetilde{w}_s(0) = b_s(0) w_s(0)\,, \tag{4.100}$$

where

$$b_s(0) = \begin{cases} 0 & \text{if the stock } s \text{ is not in the portfolio,} \\ 1 & \text{if the stock } s \text{ is in the portfolio.} \end{cases}$$

The expected portfolio return rate (4.68) can thus be written as

$$\mu_P(T) = \sum_{s=1}^{S} b_s(0) w_s(0)\, \mu_s(T) = \sum_{s=1}^{S} b_s(0)\, \mu_s^{fr}(T)\,, \tag{4.101}$$

where $\mu_s^{fr}(T)$ is the *fractional expected return rate of the stock s*. If we collect all the binary quantities $b_s(0)$ in a vector,

$$\mathbf{b}(0) = [\, b_1(0)\; b_2(0)\; \ldots\; b_S(0)\,]^T\,, \tag{4.102}$$

as well as the fractional expected return rates,

$$\boldsymbol{\mu}^{fr}(T) = [\, \mu_1^{fr}(T)\; \mu_2^{fr}(T)\; \ldots\; \mu_S^{fr}(T)\,]^T\,, \tag{4.103}$$

the expected portfolio return rate will be given by

$$\mu_P(T) = \mathbf{b}^T(0)\, \boldsymbol{\mu}^{fr}(T)\,. \tag{4.104}$$

The portfolio return rate (4.11) can be rewritten in a similar way as,

$$R_P(T) = \sum_{s=1}^{S} b_s(0) w_s(0) R_s(T) = \sum_{s=1}^{S} b_s(0) R_s^{fr}(T)\,, \tag{4.105}$$

where $R_s^{fr}(T)$ is the *fractional return rate of the stock* s. Collecting all the fractional return rates in the column vector

$$\mathbf{R}^{fr}(T) = [\, R_1^{fr}(T)\ R_2^{fr}(T)\ \ldots\ R_S^{fr}(T)\,]^T\,, \tag{4.106}$$

the portfolio return rate becomes,

$$R_P(T) = \mathbf{b}^T(0)\, \mathbf{R}^{fr}(T)\,. \tag{4.107}$$

You can easily check that the variance of the portfolio return rate has the following expression,

$$\sigma_P^2(T) = \mathbf{b}^T(0)\, \Sigma\big(\mathbf{R}^{fr}(T)\big)\, \mathbf{b}(0)\,. \tag{4.108}$$

Exercise 4.12. Complete the missing steps of the previous statement.

Any portfolio optimization problem can thus be written as,

$$\min_{\mathbf{b}(0)} \alpha\, \mathbf{b}^T(0)\, \Sigma\big(\mathbf{R}^{fr}(T)\big)\, \mathbf{b}(0)\,, \quad \text{subject to} \quad \mathbf{b}^T(0)\, \boldsymbol{\mu}^{fr}(T) = \mathcal{M}\,, \tag{4.109}$$

where $\mathbf{b}(0) \in \{0,1\}^S$.

The goal of a *binary portfolio optimization problem* is to

$$\min_{\mathbf{b}} \alpha\, \mathbf{b}^T\, \Sigma\, \mathbf{b}\,, \quad \text{subject to} \quad \boldsymbol{\mu}^T \mathbf{b} = \mathcal{M}\,, \tag{4.110}$$

where $\mathbf{b} \in \{0,1\}^S$ and Σ is a symmetric matrix. To simplify the notation we have written $\boldsymbol{\mu}^T = (\boldsymbol{\mu}^{fr})^T$. In index notation,

$$\min\, \alpha \sum_{s,s'=1}^{S} b_s\, \Sigma_{ss'}\, b_{s'}\,, \quad \text{subject to} \quad \sum_{s=1}^{S} \mu_s b_s = \mathcal{M}\,, \tag{4.111}$$

where $b_s \in \{0,1\}$ and $\Sigma_{ss'} = \Sigma_{s's}$. This is a quadratic binary optimization problem with one linear constraint. We can use the Lagrange multipliers method summarized in (3.118) to restate the problem as an unconstrained problem,

$$\begin{aligned} &\min\, \alpha \sum_{s,s'=1}^{S} b_s\, \Sigma_{ss'}\, b_{s'} - \mu\big(\sum_{s=1}^{S} \mu_s b_s - \mathcal{M}\big) \\ &\qquad = \min\, \alpha \sum_{s,s'=1}^{S} b_s\, \Sigma_{ss'}\, b_{s'} - \mu \sum_{s=1}^{S} \mu_s b_s + \mu\mathcal{M}\,. \end{aligned} \tag{4.112}$$

For consistency with (3.118), we have used μ to denote the Lagrange multiplier. However, to avoid any confusion with expected return rates that we are also denoting with the letter μ, from now on we will denote the Lagrange multiplier with the letter λ. Our optimization problem is thus,

$$\min\, \alpha \sum_{s,s'=1}^{S} b_s\, \Sigma_{ss'}\, b_{s'} - \lambda \sum_{s=1}^{S} \mu_s b_s + \lambda\mathcal{M}\,. \tag{4.113}$$

Finally, since $\lambda\mathcal{M}$ is just a constant, the problem we have to solve is basically a QUBO problem (3.163),

$$\min \alpha \sum_{s,s'=1}^{S} b_s \, \Sigma_{ss'} \, b_{s'} - \lambda \sum_{s=1}^{S} \mu_s b_s \, . \tag{4.114}$$

5 Quantum Portfolio Optimization

Now that we have reviewed the basics of quantum computing, optimization theory and modern portfolio theory, we are finally ready to present the quantum algorithm that, many believe, will help optimize investment portfolios. It is worth noting that, to this day, this algorithm has not been proved to be more efficient than the classical algorithms already in use. However, due to the computational complexity of financial problems, this algorithm as well as other similar algorithms proper of the NISQ era, are at the moment of writing the only possibilities available to us. Only future theoretical and practical results will settle the debate concerning the advantage of this algorithm.

The algorithm in question is the so called *Quantum Approximate Optimization Algorithm* (QAOA), introduced by Edward Farhi an collaborators about ten years ago. The QAOA is a hybrid quantum-classical algorithm, a topic we already discussed in Subsection 2.3. For convenience, let us resume the discussion where we left it.

5.1 The Quantum Approximate Optimization Algorithm

The present NISQ era is characterized by noisy quantum devices and a relatively small number of coherent qubits. In fact, there is a wide consensus among experts that reliable quantum computers will only be available in the long term. Because of this, scientists have proposed a new breed of algorithms known as *hybrid quantum-classical algorithms*. The basic idea of these hybrid models is to assign the most complicated part of the problem to a quantum computer and leave the rest of the optimization problem to a classical machine whose efficiency has already been proved. From the practical point of view, the advantage of these hybrid devices is that the quantum subroutine only needs a small number of coherent qubits and a shallow circuit to operate, a technological level that is expected to be reached in the near future.

The quantum approximate optimization algorithm (QAOA) is one of such algorithms. A quantum circuit is used to model the objective function we want to optimize. Moreover, the qubit that enters this circuit, the so called trial state, is also prepared by a set of quantum gates. The implementation of this quantum circuit is, from the technological viewpoint, the most difficult part of the algorithm. It not only encodes the cost function and prepares the trial state, it is also constantly updated with the new variational parameters suggested by the classical optimizer.

5.2 Portfolio Optimization via the QAOA

As we have seen in (4.114), a binary portfolio optimization problem aims at

$$\min \alpha \sum_{s,s'=1}^{S} b_s \Sigma_{ss'} b_{s'} - \sum_{s=1}^{S} \mu_s b_s \,, \tag{5.1}$$

where b_s is a binary variable, $b_s \in \{0,1\}$, and Σ is a symmetric matrix, $\Sigma_{ss'} = \Sigma_{s's}$.

According to the variational quantum method, we can minimize the objective function

$$f(b_1, \dots\ b_s\ \dots, b_S) = \alpha \sum_{s,s'=1}^{S} b_s \Sigma_{ss'} b_{s'} - \sum_{s=1}^{S} \mu_s b_s \,, \tag{5.2}$$

by constructing a parameterized Hamiltonian operator $\hat{H}$ and finding its minimum expectation value,

$$\min f(b_1, \dots\ b_s\ \dots, b_S) = \min \langle Q|\hat{H}|Q\rangle\ . \tag{5.3}$$

The vector $|Q\rangle$ belongs to a Hilbert space of dimension 2^S, $|Q\rangle \in \mathcal{H}^{2^S}$, and represents the physical state of a qubit. A possible basis for $\mathcal{H}^{2^S}$ is the computational basis $\{|b_1 \cdots b_s \cdots b_S\rangle\}$, where the entries of each vector $|b_1 \cdots b_s \cdots b_S\rangle$ are associated to their corresponding stocks ($b_s = 1$ if the stock s is in the portfolio and $b_s = 0$ if it does not). We can thus express $|Q\rangle$ as a linear superposition of these basis vectors,

$$|Q\rangle = \sum_{s=1}^{S} \alpha_{\dots\, b_s\, \dots} \,|\cdots\, b_s\, \cdots\rangle\ . \tag{5.4}$$

The construction of the Hamiltonian in terms of Pauli operators is relatively simple. For this, recall that,

$$Z\,|0\rangle = |0\rangle = (1 - 2\cdot 0)\,|0\rangle\ , \tag{5.5}$$

$$Z\,|1\rangle = -\,|1\rangle = (1 - 2\cdot 1)\,|1\rangle\ . \tag{5.6}$$

In more compact notation,

$$Z\,|b\rangle = (1 - 2b)\,|b\rangle\ . \tag{5.7}$$

We can reverse this relation,

$$b\,|b\rangle = \frac{1}{2}\,(I - Z)\,|b\rangle\ . \tag{5.8}$$

Therefore, for every vector $|b_s\rangle = |\cdots\, b_s\, \cdots\rangle$ we have that

$$b_s\,|b_s\rangle = \frac{1}{2}\,(I - Z_s)\,|b_s\rangle\ , \tag{5.9}$$

where $Z_s = I \otimes \ldots \otimes Z_s \otimes \ldots \otimes I$. The operator associated to the linear term in (5.2) is

$$\sum_{s=1}^{S} \mu_s b_s \mapsto \sum_{s=1}^{S} \frac{\mu_s}{2} I - \sum_{s=1}^{S} \frac{\mu_s}{2} Z_s\,. \tag{5.10}$$

The operator associated to the quadratic term in (5.2) is obtained in a similar fashion,

$$\begin{aligned}
\alpha \sum_{s,s'=1}^{S} b_s \Sigma_{ss'} b_{s'} \mapsto \alpha \sum_{s,s'=1}^{S} \frac{1}{2}\,(I - Z_s)\,\Sigma_{ss'}\,\frac{1}{2}\,(I - Z_{s'}) \\
= \alpha \sum_{s,s'=1}^{S} \frac{\Sigma_{ss'}}{4}\,(I - Z_{s'} - Z_s + Z_s Z_{s'}) \\
= \alpha \sum_{s,s'=1}^{S} \frac{\Sigma_{ss'}}{4} - \alpha \sum_{s,s'=1}^{S} \frac{\Sigma_{ss'}}{2}\, Z_s + \alpha \sum_{s,s'=1}^{S} \frac{\Sigma_{ss'}}{4}\, Z_s Z_{s'}\,.
\end{aligned} \tag{5.11}$$

(Remember the hermiticity of the Pauli operator Z, $Z = Z^\dagger$.)

Exercise 5.1. Compare this expression with the result of Exercise 3.48.

We conclude that the Hamiltonian operator associated with the cost function (5.2) is,

$$\hat{H} = \alpha \sum_{s,s'=1}^{S} \frac{\Sigma_{ss'}}{4} Z_s Z_{s'} - \sum_{s=1}^{S} \frac{1}{2}\Big(\alpha \sum_{s'=1}^{S} \Sigma_{ss'} - \mu_s\Big) Z_s$$

$$+ \sum_{s=1}^{S} \frac{1}{2}\Big(\alpha \sum_{s'=1}^{S} \frac{\Sigma_{ss'}}{2} - \mu_s\Big) I \,. \tag{5.12}$$

To render this expression more manageable, we write

$$\hat{H}_{ZZ} = \alpha \sum_{s,s'=1}^{S} \frac{\Sigma_{ss'}}{4} Z_s Z_{s'} \,, \qquad \hat{H}_Z = -\sum_{s=1}^{S} \frac{1}{2}\Big(\alpha \sum_{s'=1}^{S} \Sigma_{ss'} - \mu_s\Big) Z_s \,, \tag{5.13}$$

and

$$\hat{H}_I = \sum_{s=1}^{S} \frac{1}{2}\Big(\alpha \sum_{j=1}^{S} \frac{\Sigma_{ss'}}{2} - \mu_s\Big) \,. \tag{5.14}$$

That is,

$$\hat{H} = \hat{H}_{ZZ} + \hat{H}_Z + \hat{H}_I \,. \tag{5.15}$$

The first two terms define the *cost Hamiltonian*,

$$\hat{H}_C = \hat{H}_{ZZ} + \hat{H}_Z = \alpha \sum_{s,s'=1}^{S} \frac{\Sigma_{ss'}}{4} Z_s Z_{s'} - \sum_{s=1}^{S} \frac{1}{2}\Big(\alpha \sum_{s'=1}^{S} \Sigma_{ss'} - \mu_s\Big) Z_s \,. \tag{5.16}$$

Its corresponding unitary evolution operator is

$$U_C(\gamma) = e^{-i\gamma \hat{H}_C} = e^{-i\gamma \hat{H}_{ZZ}} e^{-i\gamma \hat{H}_Z} \,, \tag{5.17}$$

where γ is a positive parameter. Note that we have used the Campbell-Hausdorff formula and the fact that Pauli-Z operators commute between them. More explicitly,

$$U_C(\gamma) = \exp\Big[-i\gamma\,\alpha \sum_{s,s'=1}^{S} \frac{\Sigma_{ss'}}{4} Z_s Z_{s'}\Big] \exp\Big[i\gamma \sum_{s=1}^{S} \frac{1}{2}\Big(\alpha \sum_{s'=1}^{S} \Sigma_{ss'} - \mu_s\Big) Z_s\Big] \,. \tag{5.18}$$

At this point, we would like to know how to build the quantum gates corresponding to these unitaries. In order to do this, we just need to remember that the rotation of a single qubit around the a axis, with $a = x, y, z$, is given by (see QC1, equation (3.25)),

$$R_a(\theta_a) = e^{-i\sigma_a \theta_a/2} = \cos(\theta_a/2) I - i \sin(\theta_a/2) \sigma_a \,. \tag{5.19}$$

The gates associated to the two exponentials in (5.17) are obtained as follows. Consider first,

$$e^{-i\,\gamma \hat{H}_Z} = \exp\Big[\sum_{s=1}^{S} i\,\frac{\gamma}{2}\Big(\alpha \sum_{s'=1}^{S} \Sigma_{ss'} - \mu_s\Big) Z_s\Big]$$

$$= \bigotimes_{s=1}^{S} \exp\Big[i\,\frac{\gamma}{2}\Big(\alpha \sum_{s'=1}^{S} \Sigma_{ss'} - \mu_s\Big) Z_s\Big] \,. \tag{5.20}$$

To simplify the notation, we write

$$a_s = -\frac{1}{2}\Big(\alpha \sum_{s'=1}^{S} \Sigma_{ss'} - \mu_s\Big)\,, \tag{5.21}$$

so that

$$e^{-i\gamma \hat{H}_Z} = \bigotimes_{s=1}^{S} e^{-i\gamma a_s Z_s}\,. \tag{5.22}$$

Using the result we found in QC1, equation (4.72), this unitary is equivalent to

$$e^{-i\gamma \hat{H}_Z} = \bigotimes_{s=1}^{S} e^{-i\gamma a_s Z_s} = \bigotimes_{s=1}^{S} R_z(2\gamma a_s)\,. \tag{5.23}$$

This means that every qubit $s = 1, 2, \ldots, S$, is rotated an angle $2\gamma a_s$ about the z axis.

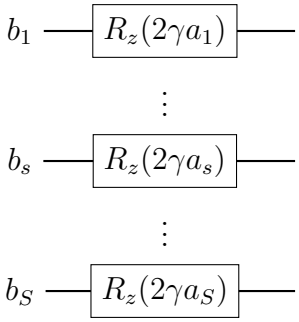

Fig. 5

The gates corresponding to the other exponential in (5.17),

$$e^{-i\gamma \hat{H}_{ZZ}} = \exp\Big[-i\gamma \sum_{s,s'=1}^{S} \frac{\alpha \Sigma_{ss'}}{4}\, Z_s Z_{s'}\Big]\,, \tag{5.24}$$

are a bit more complicated to figure out.

Exercise 5.2. Show that given two arbitrary single qubits $|b_s\rangle$ and $|b_{s'}\rangle$,

$$e^{-i\delta Z_s Z_{s'}}\,|b_s\rangle\,|b_{s'}\rangle = \mathrm{CNOT}\,(I \otimes R_z(2\delta))\,\mathrm{CNOT}\;|b_s\rangle\,|b_{s'}\rangle\;. \tag{5.25}$$

Draw the corresponding gates.

Thus, with $\alpha\Sigma_{ss'}/4 = a_{ss'}$, we conclude that

$$\begin{aligned} e^{-i\gamma \hat{H}_{ZZ}} &= \exp\Big[\sum_{s,s'=1}^{S} -i\gamma a_{ss'}\, Z_s Z_{s'}\Big] = \bigotimes_{s,s'=1}^{S} e^{-i\gamma a_{ss'}\, Z_s Z_{s'}} \\ &= \bigotimes_{\substack{s=1 \\ s'<s}}^{S} \mathrm{CNOT}_{ss'}\,(I \otimes R_z(2\gamma a_{ss'}))\,\mathrm{CNOT}_{ss'}\,. \end{aligned} \tag{5.26}$$

The corresponding gates are as follows,

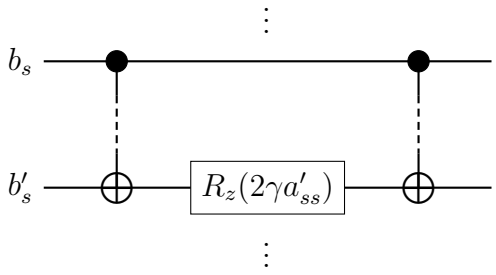

Fig. 6

Regarding the qubit that enters the circuit, it is common to choose $|Q\rangle_{in} = |+\rangle^{\otimes S}$.

Exercise 5.3. Prove that a Hadamard gate acts on a single qubit computational basis vector as

$$H\,|b\rangle = \frac{1}{\sqrt{2}}\sum_{b'}(-1)^{bb'}\,|b'\rangle\ . \tag{5.27}$$

Show that in general,

$$H^{\otimes S}\,|b_1\ldots b_S\rangle = \frac{1}{\sqrt{2^S}}\sum_{b'_s}(-1)^{b_1b'_1+\ldots+b_Sb'_S}\,|b'_1\ldots b'_S\rangle\ . \tag{5.28}$$

Thus, the initial qubit $|Q\rangle_{in} = |+\rangle^{\otimes S}$ is prepared by applying a Hadamard gate to each individual single qubit $|b_s\rangle = |0\rangle$,

$$|Q\rangle_{in} = |+\rangle^{\otimes S} = H^{\otimes S}\,|0\ldots0\rangle = \frac{1}{\sqrt{2^S}}\sum_{b'_s}|b'_1\ldots b'_S\rangle\ . \tag{5.29}$$

In this case, the state that comes out of the gate $U_C(\gamma)$ is

$$U_C(\gamma)\,|Q\rangle_{in} = U_C(\gamma)\frac{1}{\sqrt{2^S}}\sum_{b'_s}|b'_1\ldots b'_S\rangle\ . \tag{5.30}$$

Regardless of the initial qubit state we choose to enter the quantum circuit, the QAOA stipulates that we follow the unitary $U_C(\gamma)$ with a second unitary, this one associated with the so called *mixer Hamiltonian*,

$$\hat{H}_M = \sum_{s=1}^{S} X_s\,. \tag{5.31}$$

The corresponding parameterized unitary is given by

$$U_M(\beta) = e^{-i\beta\hat{H}_M} = e^{-i\beta\sum_{s=1}^{S}X_s} = \bigotimes_{s=1}^{S} e^{-i\beta X_s}\,. \tag{5.32}$$

Note that, since $U_M(\beta)$ does not commute with $U_C(\gamma)$, the strict order is $U_C(\gamma)$ followed by $U_M(\beta)$. Using the result obtained in QC1, Exercise 4.23,

$$U_M(\beta) = \bigotimes_{s=1}^{S} HR_z(2\beta)H\,. \tag{5.33}$$

Finally, since $HZH = X$,

$$U_M(\beta) = \bigotimes_{s=1}^{S} R_x(2\beta)\,. \tag{5.34}$$

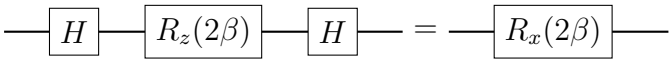

Fig. 7

The qubit state that exits the quantum circuit $U_C(\gamma)$ followed by $U_M(\beta)$, is now parameterized by the angles γ and β,

$$|Q(\gamma,\beta)\rangle = U_M(\beta)\,U_C(\gamma)\,|Q\rangle_{in}\ . \tag{5.35}$$

This is the *ansatz state* that we use to measure the cost Hamiltonian (5.16),

$$\langle Q(\gamma,\beta)|\,\hat{H}_C\,|Q(\gamma,\beta)\rangle\ . \tag{5.36}$$

Since the expectation value of the cost Hamiltonian is a real-valued function of both parameters γ and β, we write

$$F(\gamma,\beta) = \langle Q(\gamma,\beta)|\,\hat{H}_C\,|Q(\gamma,\beta)\rangle\ . \tag{5.37}$$

The measurements of the cost Hamiltonian are sent to a classical optimizer in order to propose better values for γ and β. The process is then repeated as many times as necessary until we reach a good approximation (γ^*,β^*).

Exercise 5.4. If the Hamiltonian that encodes the cost function is $\hat{H} = \hat{H}_C + \hat{H}_I$, see (5.15) and (5.16), why do we compute the expectation value of $\hat{H}_C$ instead of $\hat{H}_C + \hat{H}_I$?

The QAOA asserts that, rather than optimizing for a single pair of parameters (γ,β), a better approximation can be found if we create a sequence of unitaries $U_C(\gamma)$ and $U_M(\beta)$, each of them with its own angle parameter. That is, the QAOA prescribes that we let the initial state $|Q\rangle_{in}$ enter the following parameterized sequence of gates,

$$\begin{aligned} |Q\rangle_{in} &\to U_M(\beta_p)\,U_C(\gamma_p)\ldots U_M(\beta_k)\,U_C(\gamma_k)\ldots U_M(\beta_1)\,U_C(\gamma_1)\,|Q\rangle_{in} \\ &= |Q(\gamma_1,\ldots,\gamma_k,\ldots,\gamma_p,\beta_1,\ldots,\beta_k,\ldots,\beta_p)\rangle\ . \end{aligned} \tag{5.38}$$

Each pair of unitaries $U_M(\beta_k)U_C(\gamma_k)$, for $k = 1,2,\ldots,p$, is called a *layer*. Specifically, $U_M(\beta_k)U_C(\gamma_k)$ is the kth layer, with $U_C(\gamma_k)$ called the kth *cost layer* and $U_M(\gamma_k)$ the kth *mixer layer*. It has been proved that the larger the number of layers, the better the approximation of the optimization problem. Needless to say, the number of layers has to be kept within a reasonable limit to ensure that the calculation is not affected by excessive noise.

In total, there are $2p$ parameters to be optimized variationally: p angles γ_k and p angles β_k. We can gather all these variational parameters in a more compact

notation, $\boldsymbol{\gamma} = (\gamma_1, \ldots, \gamma_k, \ldots, \gamma_p)$ and $\boldsymbol{\beta} = (\beta_1, \ldots, \beta_k, \ldots, \beta_p)$. The *ansatz state* is thus

$$|Q_p(\boldsymbol{\gamma}, \boldsymbol{\beta})\rangle = \prod_{k=1}^{p} U_M(\beta_k)\, U_C(\gamma_k)\, |Q\rangle_{in} \ . \tag{5.39}$$

The function in $2p$ variables that the classical computer has to optimize is

$$F_p(\boldsymbol{\gamma}, \boldsymbol{\beta}) = \langle Q_p(\boldsymbol{\gamma}, \boldsymbol{\beta})|\, \hat{H}_C\, |Q_p(\boldsymbol{\gamma}, \boldsymbol{\beta})\rangle \ . \tag{5.40}$$

The optimal portfolio will thus be given by the solution of the minimization problem,

$$\min_{\boldsymbol{\gamma}, \boldsymbol{\beta}} \ F_p(\boldsymbol{\gamma}, \boldsymbol{\beta}) = \min_{\boldsymbol{\gamma}, \boldsymbol{\beta}} \ \langle Q_p(\boldsymbol{\gamma}, \boldsymbol{\beta})|\, \hat{H}_C\, |Q_p(\boldsymbol{\gamma}, \boldsymbol{\beta})\rangle \ . \tag{5.41}$$

6 Bibliography

Due to the pedagogical nature of these notes, the following bibliography is far from exhaustive. Check the references below for a complete list of original papers and comprehensive survey articles on each of the topics discussed above.

[1] K. Bharti et al, "Noisy Intermediate-Scale Quantum (NISQ) Algorithms".

[2] K. Blekos et al, "A Review on Quantum Approximate Optimization Algorithm and Its Variants".

[3] A. Bouland et al., "Prospects and Challenges of Quantum Finance".

[4] A. Canabarro et al., "Quantum Finance". In Portuguese.

[5] M. J. Capiński and E. Kopp, *Portfolio Theory and Risk Management.*

[6] M. Cerezo et al, "Variational Quantum Algorithms".

[7] G. Cornuéjols et al, *Optimization Methods in Finance.*

[8] D. J. Egger et al., "Quantum Computing for Finance".

[9] E. Farhi et al., "A Quantum Approximate Optimization Algorithm".

[10] F. Glover et al., "Quantum Bridge Analytics I: A Tutorial on Formulating and Using QUBO Models".

[11] D. A. Herman et al., "A Survey of Quantum Computing for Finance".

[12] M. S. Joshi and J. M. Paterson, *Introduction to Mathematical Portfolio Theory.*

[13] I. Kerenidis et al., "Quantum Algorithms for Portfolio Optimization".

[14] R. Orús et al., "Quantum Computing for Finance".

[15] A. L. Peressini et al., *The Mathematics of Nonlinear Programming.*

[16] C. P. Simon and L. Blume, *Mathematics for Economists.*

[17] J. P. Wheeler, *An Introduction to Optimization.*

[18] O. Zapata, "An Introduction to Quantum Computing for Physicists".

[19] O. Zapata, "Notes for a Second Course on Quantum Computing for Physicists".

Index

n qubit, 5
n-qubit gate, 6

Algorithm, 3
Ansatz state, 56, 57
Asset, 33
Attainable set, 45
Average portfolio return rate, 36

Binary digit (bit), 3
Binary knapsack problem, 32
Binary linear program, 32
Binary portfolio optimization problem, 49
Binary program, 31
Binary quadratic function, 32
Bit-flip error, 7
Bit-flip quantum repetition code, 8
Boolean (binary) model of computation , 3
Boolean algebra, 3
Boolean circuit, 4
Boolean function, 4

Circuit complexity, 4
Circuit diagram, 4
CNOT gate, 6
Computational basis states, 5
Computational error, 4, 7
Constrained continuous optimization problem, 22
Constrained integer programs, 30
Constraint, 22
Continuous optimization, 16
Convex function, 29
Convex program, Convex optimization problem, 29
Convex set, 28
Correlation coefficient, 46
Cost Hamiltonian, 53
Cost layer, 56
Covariance, 38
Covariance matrix, 39, 43
Critical point, 16, 18, 19

Decision variable, 12
Decoherence, 7
Diet problem, 26
Digital (classical) computer, 4
Discrete random variable, 37
Diversification, 47
Dual program, 29
Duality (Lagrange), 29

Efficient (optimal) portfolio, 45
Efficient circuit, 4
Efficient frontier, 45
Epigraph, 29
Equality constraint, 12
Event, 38
Expected portfolio return rate, 41
Expected stock return rate, 40
Expected value, Average, Mean, 38
Extremum, 11

Fault-tolerant quantum computer, 9
Fault-tolerant quantum era, 9
Fractional expected return rate of a stock, 48
Fractional return rate of a stock, 49

Hessian, 17
Hessian matrix, 17–19
Heuristic algorithms, 9
Hybrid quantum-classical algorithms, 9, 51
Hypograph, 45

Inefficient (suboptimal) portfolio, 45
Inequality constraint, 12
Inflection point, 10
Input, 4
Integer programs, Integer optimization problems, 30

Jacobian matrix, 21
Joint probability, 39

Knapsack problem, 30

Lagrange multiplier, 13, 24
Lagrange multipliers method, 24
Lagrange multipliers vector, 24
Lagrangian function, 13, 24
Layer, 56

Linear constraints, 27
Linear integer program, 31
Linear objective function, 27
Linear program, Linear optimization problem, 27
Logical gate (Boolean), 3
Logical gate (quantum), 4
Logical qubit, 8
Loss, 34

Maximizer, 10, 17–19
Mean-variance model, 43
Minimizer, 10, 17–19
Mixed integer linear program, 31
Mixer Hamiltonian, 55
Mixer layer, 56
Model of computation, 3
Modern portfolio theory, 33, 43

Newton method, 19
NISQ era, 9

Objective function, 11
Optimizer, 10, 17
Output, 4

Parameter (of an optimization problem), 12
Parity check, 8
Pauli gates, 5
Physical qubit, 8
Portfolio, 34
Portfolio optimization problem, 33
Portfolio price, 33
Portfolio return, 34
Portfolio return rate, 35, 41
Price vector, 35
Primal program, 29
Probability distribution, 37
Probability function, 38
Probability mass function, 37
Probability of occurrence, 37
Profit, 34

QAOA, Quantum Approximate Optimization Algorithm, 51
Quadratic objective function, 27
Quadratic program, Quadratic optimization problem, 28
Quantum advantage, 9
Quantum algorithm, 4
Quantum circuit, 4
Quantum circuit model of computation, 5
Quantum computation, 3
Quantum computer, 5
Quantum gate, 5
Quantum measurement, 6
Quantum model of computation, 4
Quantum supremacy, 9
Qubit (quantum binary digit), 4
QUBO problem, 32

Random-walk theory of stock prices, 37
Relative phase gate, Phase shift gate, 5
Return rate vector, 35
Risk, 43
Risk aversion coefficient, 43
Risk curve, 44
Risk tolerance, 43
Riskless (risk-free) asset, 43
Risky asset, 43
Rotation gate, 6

Saddle point, 17–19
Sample space, 37
Second derivative test, 10, 17–19
Set of universal quantum gates, 6
Single qubit, 5
Single-period investment model, 36
Standard deviation, 38
Steepest (gradient) descent method, 21
Step size, 21
Stock (share), 33
Stock price, 33, 39
Stock return, 34, 40
Stock return rate, 35, 40
Stock weight, 33
Strict hypograph, 45

Taylor's formula, 11, 17

Unconstrained integer program, 30

Variance, 38
Variance of a portfolio return rate, 41, 43

Variance of a stock return rate, 40

Variational quantum algorithms (VQAs), 9

Vector of expected return rates, 42
Volatility, 43

Weight vector, 35, 42