

Mahn-Soo Choi (Korea University)

A Quantum Computation Workbook

August 10, 2021

Springer

Notice

N.B. The book will be published later this year, and obviously, this draft is very rough and far from complete for the moment.

N.B. This draft is provided for educational purposes with the permission of Springer. Redistribution of this draft is strictly prohibited.

N.B. Two accompanying digital materials are freely available to the public. Both can be downloaded from the GitHub repository by clicking the links below.

- Mathematica application “Q3”: <https://github.com/quantum-mob/Q3App>
- Wolfram Language codes used to generate the demonstrations in the text:
<https://github.com/quantum-mob/QuantumWorkbook>

Preface

As is clear from the title, the book is intended for use as an introductory text and self-learning guide. It is an attempt to collect and reorganize, in a compact and integrated form, some of the fundamental principles and elementary methods in the field of quantum computation and quantum information. Properly introducing (and hence learning) quantum computation and quantum information, especially to undergraduate students, turns out to be demanding because it needs to cover subjects from various fields of science: physics of quantum mechanics, mathematics of advanced linear algebra, and computer science of information theory, to list the least. Organizing the subjects consistently in a unified language poses another layer of difficulties as each field of science usually favors its specific language.

Two digital materials accompany the book. Both are freely available at the GitHub repositories by clicking the links below.

- Mathematica application Q3 (Appendix D):
<https://github.com/quantum-mob/Q3App>
- Wolfram Language codes for the demonstrations in the book (Appendix E):
<https://github.com/quantum-mob/QuantumWorkbook>

Q3 is a Mathematica application to help study quantum information processing, quantum many-body systems, and quantum spin systems. It provides various tools and utilities for symbolic calculations and numerical simulations in these areas of quantum physics. With *Q3*, one can avoid many tedious calculations involved in various principles and theorems in quantum theory. Numerous visualization and simulation tools can help deepen the understanding of core concepts. *Quantum-Workbook* is a compilation of Mathematica Notebook files containing the Wolfram Language codes that have been used to generate the demonstrations in the book. Readers can try and modify the codes themselves to build their own examples on the demonstrations and to experiment fresh ideas. Both materials can be helpful companions through the course of self-learning for various subjects in quantum physics in general as well as in the book.

The book is a result of several years of experience in teaching quantum computation and quantum information. It has helped select and present subjects at the undergraduate level. The book aims to be self-contained. Nonetheless, it assumes some basic knowledge of quantum mechanics. Chapter 1 summarizes the fundamental postulates of quantum mechanics. It effectively provides a brief review of basic

concepts and fundamental principles in quantum mechanics. Chapter 2 presents elementary quantum gates for universal quantum computation and their properties. They are building blocks for quantum algorithms and quantum communication protocols. Chapter 3 discusses physical methods and principles to implement the elementary quantum gates. It introduces different quantum computation schemes. Chapter 4 introduces some widely known quantum algorithms. It can help grasp an idea on the so-called ‘quantum supremacy’ of quantum algorithms over classical counterparts. Chapter 5 is dedicated to decoherence effects. It introduces mathematical methods such as Kraus representation and Lindblad equation to describe the phenomena. Chapter 6 is devoted to quantum error-correction codes. It discusses basic principles for quantum error-correction procedures and discusses several examples. Chapter 7 introduces quantum information theory. It discusses distance measures for quantum information, measures for quantum entanglement degree, and entropies for information content. To maintain a coherent structure and to focus on the primary topics, we collect mathematical theories in appendices and refer to them from the main text as necessary.

The author is indebted to students in his classes for pointing out numerous mistakes and typographical errors in the manuscript. Many people have contributed to the development of Q3 by testing and actively using it. The author thanks, especially, Ha-Eum Kim, Myeongwon Lee, and Su-Ho Choi for their energetic discussions and constructive feedback in the early stage of the development of Q3. He also appreciates bug reports and valuable comments by Boris Laurent, Mi Jung So, Yeong-ho Je, and Dongni Chen.

Seoul, Korea
August 2021

Mahn-Soo Choi

Contents

1 The Postulates of Quantum Mechanics	9
1.1 Quantum States	10
1.1.1 Pure States	10
1.1.2 Mixed States	15
1.2 Time Evolution of Quantum States	22
1.2.1 Unitary Dynamics	23
1.2.2 Quantum Noisy Dynamics	26
1.3 Measurements on Quantum States	27
1.3.1 Projection Measurements	27
1.3.2 Generalized Measurements	32
2 Quantum Computation: Overview	39
2.1 Single-Qubit Gates	40
2.1.1 Pauli Gates	40
2.1.2 Hadamard Gate	44
2.1.3 Rotations	47
2.2 Two-Qubit Gates	51
2.2.1 CNOT, CZ, and SWAP	51
2.2.2 Controlled- U Gate	61
2.2.3 General Unitary Gate	67
2.3 Multi-Qubit Controlled Gates	74
2.3.1 Gray Code	75
2.3.2 Multi-Qubit Controlled-NOT	77
2.4 Universal Quantum Computation	82
2.5 Measurements	85
3 Virtual Realizations of Quantum Computers	93
3.1 Quantum Bits	94
3.2 Dynamical Scheme	96
3.2.1 Implementation of Single-Qubit Gates	97
3.2.2 Implementation of CNOT	103
3.3 Geometric/Topological Scheme	106
3.3.1 A Toy Model	107

3.3.2	Geometric Phase	112
3.4	Measurement-Based Scheme	115
3.4.1	Single-Qubit Rotations	119
3.4.2	CNOT Gate	122
3.4.3	Graph States	125
3.5	Spin-Boson Model*	127
4	Quantum Algorithms	133
4.1	Quantum Teleportation	134
4.1.1	Nonlocality in Entanglement	135
4.1.2	Implementation of Quantum Teleportation	138
4.2	Deutsch-Jozsa Algorithm & Variants	142
4.2.1	Quantum Oracle	142
4.2.2	Deutsch-Jozsa Algorithm	148
4.2.3	Bernstein-Vazirani Algorithm	150
4.2.4	Simon's Algorithm	151
4.3	Quantum Fourier Transform (QFT)	155
4.3.1	Definition and Physical Meaning	156
4.3.2	Quantum Implementation	157
4.3.3	Semiclassical Implementation	162
4.4	Quantum Phase Estimation (QPE)	166
4.4.1	Definition	166
4.4.2	Implementation	168
4.4.3	Accuracy	170
4.4.4	Simulation of von Neumann Measurement	171
4.5	Applications	173
4.5.1	The Period-Finding Algorithm	173
4.5.2	The Order-Finding Algorithm	180
4.5.3	Quantum Factorization Algorithm	182
4.5.4	Quantum Search Algorithm	182
5	Decoherence	195
5.1	Quantum Operations	196
5.1.1	Kraus Representation	198
5.1.2	Choi Isomorphism	205
5.1.3	Unitary Representation	209
5.1.4	Examples	210
5.2	Measurements as Quantum Operations	214
5.3	Quantum Master Equation	215
5.3.1	Derivation	218
5.3.2	Examples	219
5.3.3	Solution Methods	222
5.3.4	Examples Revisited	228

6 Quantum Error-Correction Codes	235
6.1 Elementary Examples: Nine-Qubit Code	236
6.1.1 Bit-Flip Errors	236
6.1.2 Phase-Flip Error	239
6.1.3 Shor's Nine-Qubit Code	241
6.2 Quantum Error Correction	245
6.2.1 Quantum Error-Correction Conditions	246
6.2.2 Discretization of errors	247
6.3 Stabilizer Formalism	248
6.3.1 Pauli Group	251
6.3.2 Properties of Stabilizers	256
6.3.3 Unitary Gates in Stabilizer Formalism	259
6.3.4 Clifford Group	260
6.3.5 Measurements in Stabilizer Formalism	266
6.3.6 Examples	268
6.4 Stabilizer Codes	274
6.4.1 Bit-Flip Code	276
6.4.2 Phase-Flip Code	277
6.4.3 Nine-Qubit Code	279
6.4.4 Five-Qubit Code	281
6.5 Surface Codes	285
6.5.1 Toric Codes	285
6.5.2 Planar Codes	291
6.5.3 Recovery Procedure	293
A Linear Algebra	299
A.1 Vectors	299
A.1.1 Vector Space	299
A.1.2 Hermitian Product	300
A.1.3 Basis	301
A.1.4 Representations	302
A.2 Linear Operators	304
A.2.1 Linear Maps	304
A.2.2 Representations	305
A.2.3 Hermitian Conjugate of Operators	306
A.3 Dirac's Bra-Ket Notation	308
A.4 Spectral Theorems	311
A.4.1 Spectral Decomposition	311
A.4.2 Functions of Operators	312
A.5 Tensor-Product Spaces	314
A.5.1 Vectors in a Product Space	314
A.5.2 Operators on a Product Space	316

B Superoperators	319
B.1 Operators as Vectors	319
B.2 Superoperators	325
B.2.1 Matrix Representation	325
B.2.2 Operator-Sum Representation	327
B.2.3 Choi Isomorphism	331
B.3 Partial Trace	333
B.4 Partial Transposition	334
C Group Theory	337
C.1 The Concept	337
C.2 Classes	341
C.3 Invariant Subgroups	342
C.4 Cosets and Factor Groups	343
C.5 Product Groups	345
D Mathematica Application Q3	347
D.1 Installation	347
D.2 Quick Start	348
E Integrated Compilation of Demonstrations	349
E.1 Installation	349
E.2 Quick Start	350
F Solutions to Select Problems	351
F.1 The Postulates of Quantum Mechanics	351
F.2 Quantum Computation: Overview	351
F.3 Quantum Computers	354
F.4 Quantum Algorithms	356
F.5 Decoherence	357
F.6 Quantum Error-Correction Codes	357
Bibliography	361
Index	369

Chapter 1

The Postulates of Quantum Mechanics

- August 6, 2021 (v1.15)

The great compilation “Elements” (see Figure 1.1) by Euclid of Alexandria in Ptolemaic Egypt circa 300 BC established a unique logical structure for mathematics, and every mathematical theory is built upon elementary axioms and definitions with propositions and proofs following. Theories in physics also take a similar structure. For example, classical mechanics is based on Sir Isaac Newton’s three laws of motion. Called “laws”, they are in fact elementary hypotheses, that is, axioms. While this is a remarkably different custom in physics compared with the mathematical counterpart, it should not be surprising to call assumptions as laws or principles because they do not only provide physical theories with logical foundation but also determine their fate whether to describe the Nature properly or have mere existence as an intellectual framework. After all, the true value of a physical science is to understand the Nature.

Embracing the wave-particle duality and the complementarity principle, quantum mechanics has been founded on the three fundamental postulates. The founders of quantum mechanics could be more ambitious to call them laws instead of plain postulates, but every single of them defies our intuition to such an extent that it sounds more natural. For an overview, here are the fundamental postulates of quantum mechanics:

Postulate 1 The *quantum state* of a system is completely described by a state vector in the Hilbert space associated with the system.

Postulate 2 The *time evolution* of a closed quantum system is governed by the Schrödinger equation.

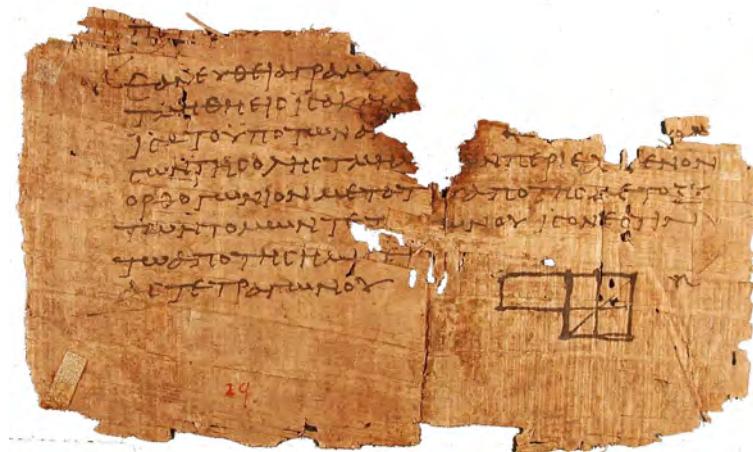


Figure 1.1: A fragment of Euclid’s Elements on part of the Oxyrhynchus papyri located at the University of Pennsylvania. Courtesy of WikiMedia Commons.

Postulate 3 A physical quantity is described by an “observable”—a Hermitian operator. Upon the *measurement* of the quantity, the outcome is one of the eigenvalues of the observable and determined *probabilistically*. Right after the measurement, the state “collapses” to the eigenstate of the observable corresponding to the measurement outcome.

In the subsequent sections of this chapter, we will detail the physical aspects of each postulate and their relevance to quantum computing and quantum information.

1.1 Quantum States

The first postulate is about how to describe the *state* of a system mathematically. Recall that in classical mechanics, the state of a particle in motion is described by the simple values of its position and momentum (or, equivalently, velocity). In quantum mechanics, the description is formulated at two different levels depending on the physical situation.

1.1.1 Pure States

Postulate 1 The quantum state of a closed system is completely described by a state vector in the Hilbert space associated with the system.

The most common example of the state vector is a “wave function”—a member of the Hilbert space of square integrable functions—originally put forward by Schrödinger. Modern approaches associate an abstract vector space with the system and the specific characters of the particular system are reflected in the

choices of basis (see Appendix A.1). When the state is known exactly, the system is said to be in the *pure state*, and the above description is comprehensive. In many cases, however, it is difficult to know the state exactly. We thus need a more general description.

This postulate immediately raises a mind blowing question: What is the physical meaning of the state vector or its components in a given basis (or the wave function)? Quantum mechanics has never offered a direct physical meaning of the state vector. It was Born (1926) who proposed a partial resolution to the question and inspired the probabilistic interpretation of quantum mechanics as formulated in Postulate 3 concerning measurement. This work awarded him the 1954 Nobel Prize in Physics.

Postulate 1 leaves another baffling question: Given a physical system, there is no general prescription to figure out the Hilbert space associated with it. While it is rather technical, it is nevertheless an important and serious question when one encounters a new (or yet-to-be-understood) system and tries to describe it quantum mechanically.

For a *qubit*—an idealistic two-level quantum system—the Hilbert space is two dimensional. A basis of two logical states $|0\rangle$ and $|1\rangle$ is assumed and is called the *logical basis* of the qubit.

Consider a group of two-level quantum systems, indicated by the symbol **s**.

Let[Qubit, s]

Different qubits can be specified by the flavor indices, the last of which has a special meaning (see the documentation of **Qubit**).

```
In[1]:= {S[1, None], S[2, None]}
S[{1, 2}, None]
Out[1]= {S1, S2}
Out[2]= {S1, S2}
```

The associated Hilbert space is two dimensional. For many functions dealing with qubits, the final index **None** can be dropped.

```
In[3]:= bs = Basis[S[1, None]]
bs = Basis[S[1]]
Out[3]= {|->, |1s1>}
Out[4]= {|->, |1s1>}
```

For the efficiency reasons, the default value 0 of any qubit is removed from the data structure. For a more intuitively appealing form with all default values, **LogicalForm** can be used.

```
In[5]:= LogicalForm[bs]
Out[5]= {|\thetas1>, |1s1>}
```

Each state in the logical basis can also be specified manually.

```
In[5]:= vec = Ket[S[1] → 1, S[2] → 0];
LogicalForm[vec, {S[1], S[2]}]
Out[5]= |1S10S2

```

```
In[6]:= vec = Ket[{S[1], S[2]} → {1, 0}];
LogicalForm[vec, {S[1], S[2]}]
Out[6]= |1S10S2

```

A general quantum state of `S[1, None]` is a linear combination of the two basis states with two complex coefficients `c[0]` and `c[1]`.

```
In[7]:= Let[Complex, c]
vec = Ket[S[1] → 0] × c[0] + Ket[S[1] → 1] × c[1];
vec // LogicalForm
Out[7]= c0 |0S11 |1S1

```

A two-dimensional state vector is often visualized as a point (called the *Bloch vector*) on the *Bloch sphere*. The Bloch sphere is a geometrical representation of a two-dimensional vector space. Any state vector $|\psi\rangle$ is expanded in the logical basis as

$$|\psi\rangle = |0\rangle \psi_0 + |1\rangle \psi_1 \quad (\psi_0, \psi_1 \in \mathbb{C}). \quad (1.1)$$

The normalization condition, $|\psi_0|^2 + |\psi_1|^2 = 1$, tells us that the state vector can be expressed up to a global phase factor by

$$|\psi\rangle = |0\rangle \cos(\theta/2) + |1\rangle \sin(\theta/2)e^{i\phi} \quad (1.2)$$

with θ and ϕ specifying the relative magnitude and phase, respectively, of the expansion coefficients ($\theta, \phi \in \mathbb{R}$). The Bloch vector associated with the state vector $|\psi\rangle$ is defined by $\mathbf{b} = (\sin \theta \cos \phi, \sin \theta \sin \phi, \cos \theta)$. The Bloch vector can equivalently be obtained in terms of the expectation values of the Pauli operators, $\mathbf{b} = (\langle \hat{\sigma}^x \rangle, \langle \hat{\sigma}^y \rangle, \langle \hat{\sigma}^z \rangle)$. Indeed, with $|\psi\rangle$ in the form (1.2), one can show that $\langle \hat{\sigma}^x \rangle = \sin \theta \cos \phi$, $\langle \hat{\sigma}^y \rangle = \sin \theta \sin \phi$, and $\langle \hat{\sigma}^z \rangle = \cos \theta$. Therefore, any state vector in a two-dimensional Hilbert space corresponds uniquely (up to a global phase factor) to a point on the sphere of unit radius, the Bloch sphere.

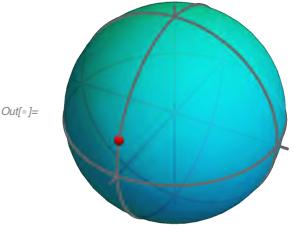
A two-dimensional pure state is represented by a point on the Bloch sphere. For example, consider a pure state.

```
In[8]:= vec = Ket[] × Sqrt[2] - I Ket[S[1] → 1];
vec // LogicalForm
Out[8]= √2 |0S11

```

This visualize the state vector on a Bloch sphere. `BlochVector` converts the state vector to a three-dimensional vector. `BlochSphere` is a shortcut for `Graphics3D` with an visualization of the Bloch sphere.

```
In[=]:= BlochSphere[{Red, Bead@BlochVector[vec]},  
ImageSize -> Small]
```



Many quantum systems like a system of many particles are composed of several parts with independent degrees of freedom. For such a system, the overall Hilbert space is built up from the Hilbert spaces of individual parts by means of the tensor product (see Appendix A.5). For example, consider a system of two parts and suppose that they are associated with the vector spaces \mathcal{V} and \mathcal{W} , respectively. The total Hilbert space is given by the tensor product $\mathcal{V} \otimes \mathcal{W}$, which is defined to be the vector space spanned by the tensor-product basis

$$\{|v_i\rangle \otimes |w_j\rangle : i = 0, \dots, m-1; j = 0, \dots, n-1\}, \quad (1.3)$$

where $\{|v_i\rangle\}$ and $\{|w_j\rangle\}$ are bases of \mathcal{V} and \mathcal{W} of dimensions m and n , respectively. The dimension of $\mathcal{V} \otimes \mathcal{W}$ for the total system is obviously given by mn , and in general a state vector of the total system is a linear superposition consisting of mn terms

$$|\Psi\rangle = \sum_{i=0}^{m-1} \sum_{j=0}^{n-1} |v_i\rangle \otimes |w_j\rangle \Psi_{ij}. \quad (1.4)$$

Some state vectors are factored into the form

$$(|v_1\rangle c_1 + |v_2\rangle c_2 + \dots) \otimes (|w_1\rangle d_1 + |w_2\rangle d_2 + \dots). \quad (1.5)$$

Such a state is said to be *separable*. For example, the superposition of all the logical basis states of two qubits

$$|0\rangle \otimes |0\rangle + |0\rangle \otimes |1\rangle + |1\rangle \otimes |0\rangle + |1\rangle \otimes |1\rangle \quad (1.6)$$

is factored into

$$(|0\rangle + |1\rangle) \otimes (|0\rangle + |1\rangle), \quad (1.7)$$

and is a separable state. On the other hand, the state

$$|0\rangle \otimes |0\rangle + |0\rangle \otimes |1\rangle + |1\rangle \otimes |0\rangle \quad (1.8)$$

can never be factored. Such a state is said to be *entangled* (Fig. 1.2). The *Schmidt decomposition* is a systematic way to test whether a given quantum state $|\Psi\rangle$ in a tensor-product space is separable or not. The Schmidt decomposition—see Appendix A.5.1 for the mathematical details—is a method to choose proper bases

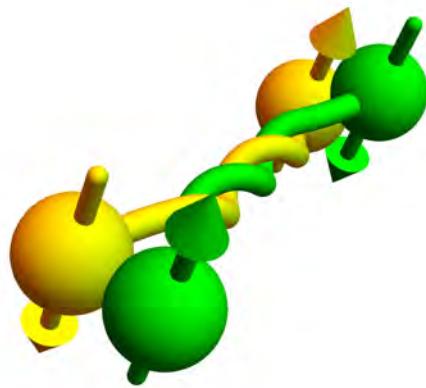


Figure 1.2: An artist’s view of quantum entanglement. The up and down arrows represent the logical basis states $|0\rangle$ and $|1\rangle$.

$\{|v'_i\rangle\}$ and $\{|w'_j\rangle\}$ of \mathcal{V} and \mathcal{W} , respectively, to rewrite the given state vector $|\Psi\rangle$ in the least number of terms of the form

$$|\Psi\rangle = \sum_{k=0}^{R-1} |v'_k\rangle \otimes |w'_k\rangle s_k, \quad (1.9)$$

where the coefficients are all definitely positive, $s_k > 0$. Here the so-called *Schmidt rank* R of the quantum state $|\Psi\rangle$ cannot be larger than the smaller of m and n , $R \leq \min(m, n)$. If the Schmidt rank is 2 or larger, then the state is entangled.

Consider the following state of a two-qubit state.

```
In[1]:= ket = Ket[] + Ket[S[1] → 1] + Ket[S[2] → 1];
ket // LogicalForm
Out[1]= |0S1 0S2⟩ + |0S1 1S2⟩ + |1S1 0S2⟩
```

This gives the Schmidt decomposition of the state. It turns out that its Schmidt rank is two and the state is an entangled state.

```
In[2]:= {ww, uu, vv} = SchmidtDecomposition[ket, S[1], S[2]];
ww
Out[2]= {√(1/2 × (3 + √5)), √(1/2 × (3 - √5))}
```

SchmidtForm presents the Schmidt decomposition in a more intuitively-appealing form. For a thorough analysis of the result, use **SchmidtDecomposition**.

```
In[5]:= new = SchmidtForm[ket, {S[1]}, {S[2]}]
Out[5]= 
$$\sqrt{\frac{1}{2} \times (3 - \sqrt{5})} \left( \begin{array}{c} \left(1 + \frac{1}{2} \times (1 - \sqrt{5})\right) |0_{S_1}\rangle \\ \sqrt{\frac{1}{4} (1 - \sqrt{5})^2 + \left(1 + \frac{1}{2} \times (1 - \sqrt{5})\right)^2} \end{array} \right) \otimes \left( \begin{array}{c} (1 - \sqrt{5}) |1_{S_1}\rangle \\ 2 \sqrt{\frac{1}{4} (1 - \sqrt{5})^2 + \left(1 + \frac{1}{2} \times (1 - \sqrt{5})\right)^2} \end{array} \right)$$


$$\left( \begin{array}{c} (1 - \sqrt{5}) |0_{S_2}\rangle \\ 2 \sqrt{1 + \frac{1}{4} (1 - \sqrt{5})^2} \end{array} \right) + \sqrt{\frac{1}{2} \times (3 + \sqrt{5})} \left( \begin{array}{c} |1_{S_2}\rangle \\ \sqrt{1 + \frac{1}{4} (1 - \sqrt{5})^2} \end{array} \right)$$


$$\left( \begin{array}{c} \left(1 + \frac{1}{2} \times (1 + \sqrt{5})\right) |0_{S_1}\rangle \\ \sqrt{\frac{1}{4} (1 + \sqrt{5})^2 + \left(1 + \frac{1}{2} \times (1 + \sqrt{5})\right)^2} \end{array} \right) \otimes \left( \begin{array}{c} (1 + \sqrt{5}) |1_{S_1}\rangle \\ 2 \sqrt{\frac{1}{4} (1 + \sqrt{5})^2 + \left(1 + \frac{1}{2} \times (1 + \sqrt{5})\right)^2} \end{array} \right)$$


$$\left( \begin{array}{c} (1 + \sqrt{5}) |0_{S_2}\rangle \\ 2 \sqrt{1 + \frac{1}{4} (1 + \sqrt{5})^2} \end{array} \right) + \frac{|1_{S_2}\rangle}{\sqrt{1 + \frac{1}{4} (1 + \sqrt{5})^2}}$$

```

The Schmidt decomposition is incredibly complicated for such a simple-looking system. Let us take an approximation to get an impression of how the state is entangled.

```
In[6]:= new // N // Simplify
Out[6]= 0.618034 (0.525731 |0_{S_1}\rangle - 0.850651 |1_{S_1}\rangle) \otimes (-0.525731 |0_{S_2}\rangle + 0.850651 |1_{S_2}\rangle) +
1.61803 (0.850651 |0_{S_1}\rangle + 0.525731 |1_{S_1}\rangle) \otimes (0.850651 |0_{S_2}\rangle + 0.525731 |1_{S_2}\rangle)
```

As pointed out for the first time by [Einstein *et al.* \(1935\)](#), entangled quantum states have many intriguing properties that are difficult to understand intuitively and have raised many questions concerning the foundation of quantum mechanics. The non-local property—the very property pointed out by [Einstein *et al.* \(1935\)](#)—is the representative property of entangled states, and illustrated in Section 4.1.1. Quantum entanglement also provides a fundamental explanation for quantum decoherence—the process where the quantum system loses the quantum effects—as is discussed in Chapter 5. In quantum computation, quantum information and quantum communication, quantum entanglement is regarded as a valuable resource that enables many amazing quantum effects such quantum speed-up and unconditional security which are impossible in classical world. One of the most illustrative example—the quantum teleportation—is discussed in Section 4.1.

1.1.2 Mixed States

One often encounters a situation where the state of a system is not known completely. Common example is the case where the system is interacting with its surroundings. In this case, the system is said to be in a *mixed state*. A mixed state is a statistical mixture of pure states and characterized in terms of statistical ensemble, where different state vectors $|\psi_\mu\rangle$ are found with probabilities p_μ .^{1.1}

^{1.1}Here, note that the different pure states $|\psi_\mu\rangle$ in the mixture do not have to be orthogonal to each other. They do not span the Hilbert space, either, in general. They are completely general.

Such an ensemble is efficiently represented by a *density operator* (often just called *density matrix* for historical reasons) which is constructed as

$$\hat{\rho} = \sum_{\mu} |\psi_{\mu}\rangle p_{\mu} \langle \psi_{\mu}| \quad (1.10)$$

Sometimes, it is convenient to describe the statistical mixture in terms of a set of *unnormalized* vectors $\{|\psi'_j\rangle := |\psi_{\mu}\rangle \sqrt{p_{\mu}}\}$ absorbing the probabilities into the vectors— $\langle \psi'_{\mu} | \psi'_{\mu} \rangle$ gives the probability to find the state $|\psi'_{\mu}\rangle$ in the ensemble. The corresponding density operator is constructed as

$$\hat{\rho} = \sum_{\mu} |\psi'_{\mu}\rangle \langle \psi'_{\mu}|, \quad (1.10')$$

and is certainly equivalent to the one in (1.10).

Consider a density operator representing a statistical mixture of two pure states.

```
In[1]:= vecs = {Ket[], (Ket[] - I Ket[S[1] -> 1]) / Sqrt[2]};
vecs // LogicalForm
prbs = {1/3, 2/3}
Out[1]= {|\Theta_{S_1}\rangle, \frac{|\Theta_{S_1}\rangle - i|\mathbf{1}_{S_1}\rangle}{\sqrt{2}}}
Out[2]= \left\{ \frac{1}{3}, \frac{2}{3} \right\}
```

From the specifications of the ensemble, this constructs the density operator for the mixed state.

```
In[3]:= \rho = (vecs ** Dagger[vecs]).prbs // Garner;
\rho // LogicalForm
Out[3]= \frac{2}{3} |\Theta_{S_1}\rangle \langle \Theta_{S_1}| + \frac{1}{3} i |\Theta_{S_1}\rangle \langle \mathbf{1}_{S_1}| - \frac{1}{3} i |\mathbf{1}_{S_1}\rangle \langle \Theta_{S_1}| + \frac{1}{3} |\mathbf{1}_{S_1}\rangle \langle \mathbf{1}_{S_1}|
```

This gives the matrix representation -- the “density matrix” -- of the density operator in the logical basis.

```
In[4]:= Matrix@{\rho} // MatrixForm
Out[4]/MatrixForm=
\begin{pmatrix} \frac{2}{3} & \frac{i}{3} \\ \frac{-i}{3} & \frac{1}{3} \end{pmatrix}
```

This gives the expression of the density operator in terms of the Pauli operators.

```
In[5]:= Elaborate@ExpressionFor[Matrix@{\rho}, S[1]]
Out[5]= \frac{1}{2} - \frac{S_1^y}{3} + \frac{S_1^z}{6}
```

By construction, any density operator $\hat{\rho}$ is Hermitian, $\hat{\rho}^\dagger = \hat{\rho}$. Obviously, a density operator is an operator acting on the state vectors in the vector space \mathcal{H} . However, it is more appropriate to regard it as a vector in the vector space $\mathcal{L}(\mathcal{H})$ of all linear operators on \mathcal{H} (see Appendix B.1).

Each diagonal element $\rho_{jj} := \langle v_j | \hat{\rho} | v_j \rangle$ in a given basis $\{|v_j\rangle\}$ carries an important physical meaning. ρ_{jj} gives the probability P_j to find the system in the basis state $|v_j\rangle$. To see it, note that the probability to find the system in $|v_j\rangle$ is $|\langle v_j | \psi_\mu \rangle|^2$ under the condition that the state is $|\psi_\mu\rangle$. The latter has the chance of probability p_μ . Therefore the overall probability P_j is given by

$$P_j = \sum_\mu |\langle v_j | \psi_\mu \rangle|^2 p_\mu = \sum_\mu \langle v_j | \psi_\mu \rangle p_\mu \langle \psi_\mu | v_j \rangle = \langle v_j | \hat{\rho} | v_j \rangle \quad (1.11)$$

as expected. The physical meaning of diagonal elements further implies several basic yet important properties of density operator (see Problems 1.1 and 1.2): (i) A density operator is not only Hermitian but also *positive* (Definition A.12). (ii) A density operator has the unit trace, $\text{Tr } \hat{\rho} = 1$. (iii) Any eigenvalue of a density operator $\hat{\rho}$ lies between 0 and 1—it is often said that $0 < \hat{\rho} \leq 1$.

On the other hand, off-diagonal elements of a density operator are responsible for coherence effects as we will observe in various interference experiments later. It is important to note here that coherence is a basis-dependent effect.

It is interesting to note that the density operator and the relevant physical properties do not depend on all the details in the specification of the statistical ensemble, which in some sense makes the description of mixed states in terms of density operator powerful and efficient: Suppose that two statistical ensembles be specified by the sets $\{|\alpha_\mu\rangle : \mu = 1, \dots, m\}$ and $\{|\beta_\nu\rangle : \nu = 1, \dots, n\}$, respectively, of unnormalized vectors as in Eq. (1.10'). Without loss of generality, assume that $m \leq n$. Then, the density operators describing the ensembles are identical, that is,

$$\sum_{\mu=1}^m |\alpha_\mu\rangle \langle \alpha_\mu| = \sum_{\nu=1}^n |\beta_\nu\rangle \langle \beta_\nu| \quad (1.12)$$

if and only if there exists an $n \times n$ unitary matrix $U_{\mu\nu}$ such that

$$|\beta_\nu\rangle = \sum_\mu |\alpha_\mu\rangle U_{\mu\nu} \quad (1.13)$$

for all $\nu = 1, \dots, n$. Although providing a rigorous proof is out of the main scope of the book, the impact of the unitary freedom—or ambiguity—in the specification of the mixed states is wide spread. Further, similar unitary freedom is observed in the description of decoherence and related effects (Section 5). It is thus heuristic to take a moment here to prove it.

It is clear that if the states from the two sets satisfy the relation (1.13), then the identity (1.12) holds. To see the converse, suppose that the two density operators are the same and equal to $\hat{\rho}$. Write $\hat{\rho}$ in a spectral decomposition (Appendix A.4),

$$\hat{\rho} = \sum_\lambda |\lambda\rangle \langle \lambda|, \quad (1.14)$$

where $|\lambda\rangle$ are the (unnormalized) eigenstates of $\hat{\rho}$ belonging to non-zero eigenvalues λ . We have used the properties (Theorem A.19) of positive operators and normalized $|\lambda\rangle$ by their own eigenvalues, that is, $\langle\lambda|\lambda\rangle = \lambda$. We first note that $\{|\alpha_\mu\rangle\}$ and $\{|\lambda\rangle\}$ span the same subspace, and hence $|\alpha_\mu\rangle$ can be expanded in $|\lambda\rangle$,

$$|\alpha_\mu\rangle = \sum_{\lambda} |\lambda\rangle V_{\lambda\mu}. \quad (1.15)$$

Putting it into (1.12), we require that

$$\sum_{\lambda\lambda'} |\lambda\rangle \langle\lambda'| \sum_{\mu} V_{\lambda\mu} V_{\lambda'\mu}^* = \sum_{\lambda} |\lambda\rangle \langle\lambda|. \quad (1.16)$$

Recall that unlike $|\alpha_\mu\rangle$, which are not orthogonal to each other in general, $|\lambda\rangle$ are orthogonal (Appendix A.4). Equation (1.16) thus implies that the rows of the matrix V are orthogonal to each other. By adding additional rows, if necessary, that are orthogonal to existing rows, the matrix V can be extended into a unitary matrix. For the same reason, $|\beta_\nu\rangle$ can be expanded as

$$|\beta_\nu\rangle = \sum_{\lambda} |\lambda\rangle W_{\lambda\nu}, \quad (1.17)$$

and the matrix $W_{\lambda\nu}$ can be extended into a unitary matrix. Overall, $|\alpha_\mu\rangle$ and $|\beta_\nu\rangle$ satisfy the relation (1.13), where $U = W^\dagger V$.^{1.2} Finally, here is a demonstration of such freedom:

Consider a statistical mixture of the following three pure states.

```
In[7]:= v1 = Ket[];
v2 = (Ket[] + I Ket[S -> 1]) / Sqrt[2];
v3 = (Ket[] 2 + Ket[S -> 1] I) / Sqrt[5];
LogicalForm@{v1, v2, v3}
Out[7]= { |0s>, (|0s> + i |1s>) / Sqrt[2], (2 |0s> + i |1s>) / Sqrt[5] }
```

These are the associated probabilities.

```
In[8]:= p1 = 1 / 8;
p2 = 1 / 4;
p3 = 5 / 8;
{p1, p2, p3}
Out[8]= {1/8, 1/4, 5/8}
```

The mixed state is described by the density operator.

^{1.2}In this proof, we have implicitly assumed that $m = n$. However, the proof can be extended easily by adding null vectors in the set $\{\alpha_\mu\}$ until $m = n$.

```
In[=]:= ρ = Total@Multiply[{v1, v2, v3}, {p1, p2, p3}, Dagger@{v1, v2, v3}];  
ρ // LogicalForm  
Out[=]=  $\frac{3 |\Theta_S\rangle \langle \Theta_S|}{4} - \frac{3 i}{8} |\Theta_S\rangle \langle 1_S| + \frac{3 i}{8} |\Theta_S\rangle \langle 1_S| + \frac{|\Theta_S\rangle \langle 1_S|}{4}$ 
```

Next consider another set of pure states.

```
In[=]:= w1 = (Ket[] 2 + Ket[S → 1] I) / Sqrt[5];  
w2 = Ket[S → 1];  
LogicalForm@{w1, w2}  
Out[=]=  $\left\{ \frac{2 |\Theta_S\rangle + i |1_S\rangle}{\sqrt{5}}, |1_S\rangle \right\}$ 
```

The associated probabilities are as following.

```
In[=]:= q1 = 15 / 16;  
q2 = 1 / 16;  
{q1, q2}  
Out[=]=  $\left\{ \frac{15}{16}, \frac{1}{16} \right\}$ 
```

The mixture leads to the same density operator.

```
In[=]:= σ = Total@Multiply[{w1, w2}, {q1, q2}, Dagger@{w1, w2}];  
Out[=]=  $\frac{3 |\_\rangle \langle \_|}{4} - \frac{3 i |\_\rangle \langle 1_S|}{8} + \frac{3 i |\Theta_S\rangle \langle \_|}{8} + \frac{|\Theta_S\rangle \langle 1_S|}{4}$ 
```

The two sets are related by the following unitary matrix

```
In[=]:= U = Topple@{  
{1, 1, 2} / Sqrt[6],  
{1, -1, 0} / Sqrt[2],  
{1, 1, -1} / Sqrt[3]  
};  
U // MatrixForm  
Out[=]//MatrixForm=  $\begin{pmatrix} \frac{1}{\sqrt{6}} & \frac{1}{\sqrt{2}} & \frac{1}{\sqrt{3}} \\ \frac{1}{\sqrt{6}} & -\frac{1}{\sqrt{2}} & \frac{1}{\sqrt{3}} \\ \sqrt{\frac{2}{3}} & 0 & -\frac{1}{\sqrt{3}} \end{pmatrix}$ 
```

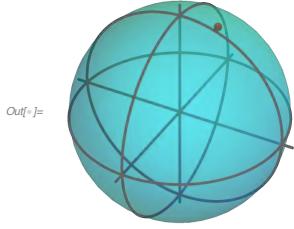
Recall that two-dimensional pure states are visualized *on* a Bloch sphere. A mixed state $\hat{\rho}$ for a qubit can be visualized similarly, but in general the representing point resides *inside* the Bloch sphere: As for a pure state, the Bloch vector \mathbf{b} corresponding to a mixed state $\hat{\rho}$ is defined by $\mathbf{b} = (\langle \hat{\sigma}^x \rangle, \langle \hat{\sigma}^y \rangle, \langle \hat{\sigma}^z \rangle)$. Recalling that any operator on a two-dimensional vector space is a linear superposition of the Pauli operators, we decompose a density operator into the form

$$\hat{\rho} = \frac{1}{2} (\hat{\sigma}^0 + x\hat{\sigma}^x + y\hat{\sigma}^y + z\hat{\sigma}^z) \quad (1.18)$$

with $x, y, z \in \mathbb{R}$. By evaluating the averages, $\langle \hat{\sigma}^\mu \rangle = \text{Tr } \hat{\rho} \hat{\sigma}^\mu$, of the Pauli operators with respect to $\hat{\rho}$, one can see that the Bloch vector corresponding to $\hat{\rho}$ is just given by $\mathbf{b} = (x, y, z)$. Clearly, $\sqrt{x^2 + y^2 + z^2} \leq 1$, and it may lie inside the Bloch sphere in general.

This gives a visualization of the mixed state by a point -- Bloch vector -- in a Bloch sphere.

```
In[1]:= BlochSphere[{Red, Bead@BlochVector@ρ}, "Opacity" → 0.4, ImageSize → Small]
```



A mixed state arises naturally when the system interacts with its environment: As a closed system, the total system is described by a pure state $|\Psi\rangle \in \mathcal{H} \otimes \mathcal{E}$, where \mathcal{H} and \mathcal{E} are the Hilbert spaces associated with the system and the environment, respectively. In accordance with the *Schmidt decomposition* (see Appendix A.5), it can be written as

$$|\Psi\rangle = \sum_{j=1}^R |\alpha_j\rangle \otimes |\beta_j\rangle \sqrt{p_j}, \quad (1.19)$$

where $|\alpha_j\rangle \in \mathcal{H}$, $|\beta_j\rangle \in \mathcal{E}$, $0 < p_j < 1$, and R is the *Schmidt rank* of $|\Psi\rangle$. Without access to the environment, one cannot tell in which state among $|\alpha_j\rangle$ the system is. One can only tell the chances. The probability for the system to be found in $|\alpha_j\rangle$ is equal to p_j . Therefore, the density operator that describes the situation is given by

$$\hat{\rho} = \sum_j |\alpha_j\rangle \langle \alpha_j| p_j \quad (1.20)$$

Now noting that $\langle \beta_i | \beta_j \rangle = \delta_{ij}$ and

$$|\Psi\rangle \langle \Psi| = \sum_{ij} |\alpha_i\rangle \langle \alpha_j| \otimes |\beta_i\rangle \langle \beta_j| \sqrt{p_i p_j}, \quad (1.21)$$

we can see that the expression in (1.20) is equivalent to taking a partial trace over \mathcal{E} (see Appendix B.3),

$$\hat{\rho} = \text{Tr}_{\mathcal{E}} |\Psi\rangle \langle \Psi|. \quad (1.22)$$

In this sense, taking a partial trace over a particular part of the total system corresponds physically to “ignoring” that part.

Suppose that two qubits are coupled and that the total system is in the following state. We can regard the first qubit as the “system” and the second as the “reservoir”.

```
In[2]:= total = (Ket[] - Ket[S[1] → 1] + Ket[S[2] → 1]) / Sqrt[3];  
LogicalForm[total]
```

Out[2]=

$$\frac{|\theta_{S_1} 0_{S_2}\rangle + |\theta_{S_1} 1_{S_2}\rangle - |1_{S_1} 0_{S_2}\rangle}{\sqrt{3}}$$

The first qubit is in a mixed state. The density operator is given by the partial trace over the second qubit.

```
In[7]:= ρ = Elaborate@PartialTrace[total ** Dagger[total], S[2]]
Out[7]=  $\frac{1}{2} - \frac{S_1^x}{3} + \frac{S_1^z}{6}$ 
```

This is the matrix representation of the density operator in the logical basis.

```
In[8]:= MatrixForm@Matrix@ρ
Out[8]//MatrixForm=

$$\begin{pmatrix} \frac{2}{3} & -\frac{1}{3} \\ \frac{1}{3} & \frac{1}{3} \\ -\frac{1}{3} & \frac{1}{3} \end{pmatrix}$$

```

A simple yet important aspect of the above observation is that when $|\Psi\rangle$ of the total system is a separable state, $|\Psi\rangle = |\alpha\rangle \otimes |\beta\rangle$, the reduced state

$$\hat{\rho} = \sum_{\mathcal{E}} |\alpha\rangle \langle \alpha| \otimes |\beta\rangle \langle \beta| = |\alpha\rangle \langle \alpha| \quad (1.23)$$

is a pure state, retaining full coherence. It means that the *entanglement* between the system and the environment is responsible of decoherence. In the history of quantum physics, it was a mystery why quantum effects can only be observable in microscopic systems but not in macroscopic systems. It was entanglement that has provided a key clue to the resolution of the mystery. We will see later entanglement also provides valuable resources for quantum information processing and quantum communications—see, for example, Section 4.1.

When the system is in a pure state $|\psi\rangle$, the density operator is simply given by $\hat{\rho} = |\psi\rangle \langle \psi|$. For a pure state $|\psi\rangle = |0\rangle c_0 + |1\rangle c_1$, the matrix representation of $\hat{\rho}$ is given by

$$\hat{\rho} \doteq \begin{bmatrix} c_0 c_0^* & c_0 c_1^* \\ c_1 c_0^* & c_1 c_1^* \end{bmatrix}. \quad (1.24)$$

Given a density operator $\hat{\rho}$, there is a simple test whether the corresponding state is a pure state or a mixed state: It is a pure state if $\text{Tr } \hat{\rho}^2 = 1$; see Problem 1.3. The converse is trivially true. For a mixed state, $\text{Tr } \hat{\rho}^2$ is strictly smaller than 1.

Let us examine the density operator corresponding to a pure state. As an example, consider the following pure state.

```
In[8]:= vec = Ket[S[1] → 0] × c[0] + Ket[S[1] → 1] × c[1];
vec // LogicalForm
Out[8]= c₀ |0s₁⟩ + c₁ |1s₁⟩
```

This gives the density operator corresponding to the pure state.

```
In[9]:= ρ = vec ** Dagger[vec];
ρ // LogicalForm
Out[9]= c₀ c₀ |0s₁⟩ ⟨0s₁| + c₀ c₁ |0s₁⟩ ⟨1s₁| + c₁ c₀ |1s₁⟩ ⟨0s₁| + c₁ c₁ |1s₁⟩ ⟨1s₁|
```

The matrix representation of the density operator gives the typical form of the density matrix for a pure state.

```
In[=]:= mat = Matrix[\rho];
mat // MatrixForm
Out[=]//MatrixForm=

$$\begin{pmatrix} c_0 & c_0^* & c_0 & c_1^* \\ c_1 & c_0^* & c_1 & c_1^* \end{pmatrix}$$

This illustrates that  $\text{Tr}[\rho^2] = 1$  for pure states.

In[=]:= Tr[mat.mat] // Simplify // MatrixForm
Out[=]//MatrixForm=

$$(c_0 c_0^* + c_1 c_1^*)^2$$

```

A far more general way to characterize a mixed state $\hat{\rho}$ is the *von Neumann entropy*. The von Neumann entropy of a density operator $\hat{\rho}$ is defined by

$$S(\hat{\rho}) := -\text{Tr } \hat{\rho} \log_2 \hat{\rho} = -\sum_j \lambda_j \log_2 \lambda_j, \quad (1.25)$$

where λ_j are the non-vanishing eigenvalues of $\hat{\rho}$. It is an extension of the notion of classical Shannon entropy. The base of the logarithm in (1.25) is arbitrary. It is customary and convenient to take the logarithm of base 2 in quantum information theory. The von Neumann entropy of any pure state is zero because the only eigenvalue of the density operator representing a pure state is 1. It is $\log_2 N$ for the completely random state, $\hat{\rho} = \hat{I}/N$, where N is the dimension of the Hilbert space. In fact, $\log_2 N$ is the maximum that $S(\hat{\rho})$ can take for any $\hat{\rho}$. The von Neumann entropy measures the uncertainty about the quantum states in the statistical mixture associated with the density operator.

As a vector describing a mixed state, one can ask whether a density operator is separable or not. Consider a system consisting of two subsystems A and B . A density operator $\hat{\rho}$ (and the associated mixed state) is said to be separable if it can be written as a *convex linear combination*

$$\hat{\rho} = \sum_j \hat{\sigma}_j \otimes \hat{\tau}_j p_j, \quad 0 \leq p_j \leq 1, \quad \sum_j p_j = 1, \quad (1.26)$$

where $\hat{\sigma}_j$ and $\hat{\tau}_j$ are the density operators of the subsystems A and B . Unlike pure states, for which the Schmidt decomposition provides a simple test of entanglement (see Appendix A.5), it is hard in general and remains an open question to tell whether a given mixed state is separable or entangled. One might be tempted to use the Schmidt-like decomposition (A.57) of operators for the test of mixed-state entanglement. However, the operators \hat{A}_μ and \hat{B}_μ in the sum are not guaranteed to be density operators.

1.2 Time Evolution of Quantum States

The state of a classical system changes with time in accordance with Newton's second law of motion, that is, the evolution is governed by the celebrated equation of motion, $F = ma$, by Newton. In quantum mechanics, Newton's equation of motion is replaced with Schrödinger's.

1.2.1 Unitary Dynamics

Postulate 2 The time evolution of the state $|\psi\rangle$ of a *closed* quantum system is governed by the Schrödinger equation

$$i\hbar \frac{d}{dt} |\psi\rangle = \hat{H} |\psi\rangle , \quad (1.27)$$

where \hbar is the Planck constant and \hat{H} is the Hamiltonian of the system.

Throughout the book, we will use a unit system where $\hbar = 1$. In such a unit system, energy and frequency have the same dimension, the inverse of the dimension of time. The Hamiltonian is a Hermitian operator, physically, describing the energy of the system. In general, it is difficult to find the precise Hamiltonian for a particular system. In most cases, a model Hamiltonian is constructed and tested against experimental observations. If desired, it is corrected by changing the existing terms or including additional terms, and tested again.

Postulate 2 expresses the time evolution in terms of a differential equation. An equivalent way is to describe it by means of unitary transformation: Suppose that the system is initially in a state $|\psi(0)\rangle$, then the state $|\psi(t)\rangle$ at later time $t > 0$ is related to the initial state $|\psi(0)\rangle$ by a unitary operator $\hat{U}(t)$

$$|\psi(t)\rangle = \hat{U}(t) |\psi(0)\rangle . \quad (1.28)$$

The unitary operator $\hat{U}(t)$ is called the *time-evolution operator*. When the Hamiltonian \hat{H} of the system is independent of time, the time-evolution operator is given by (recall that we have put $\hbar = 1$)

$$\hat{U}(t) = \exp[-it\hat{H}] . \quad (1.29)$$

The best way to evaluate the exponential function of a normal operator is to use its *spectral decomposition* (Appendix A.3). With the eigenstates $|E\rangle$ and corresponding eigenvalues E of \hat{H} , the Hamiltonian itself reads as $\hat{H} = \sum_E |E\rangle \langle E| E$ and hence its exponential function is given by $\exp(-it\hat{H}) = \sum_E |E\rangle \langle E| e^{-itE}$. In general, especially when the system is driven externally, for example, to actively process quantum information, the Hamiltonian depends on time. In this case, the relation between the time-evolution operator and the Hamiltonian is more complicated and will be discussed later.

Consider a two - level quantum state, denoted by the symbol S . Some real parameters will be denoted by the symbol B .

```
Let[Qubit, S]
Let[Real, B]
```

A time-independent Hamiltonian can be expressed in terms of the Pauli operators.

$$Out[=] := B_0 + B_1 S^x + B_2 S^y + B_3 S^z$$

In this case, the time-evolution operator is given by

```
In[6]:= Clear[U]
U[t_] = MultiplyExp[-I t H]
Out[6]=  $e^{-i t (B_0 + B_1 S^x + B_2 S^y + B_3 S^z)}$ 
```

The exponential function of operators can be evaluated by means of the spectral decomposition. Q3 has an internal mechanism to facilitate the spectral decomposition method. It is implemented through the function `Elaborate`.

```
In[1]:= Elaborate[U[t]] // ExpToTrig // Garner
```

```
Out[1]= 
$$\frac{\cos\left[t \sqrt{B_1^2 + B_2^2 + B_3^2}\right] \times (\cos[t B_0] - i \sin[t B_0]) - i B_3 S^z (\cos[t B_0] - i \sin[t B_0]) \times \sin\left[t \sqrt{B_1^2 + B_2^2 + B_3^2}\right]}{\sqrt{B_1^2 + B_2^2 + B_3^2}} -$$


$$\frac{i (B_1 - i B_2) S^+ (\cos[t B_0] - i \sin[t B_0]) \times \sin\left[t \sqrt{B_1^2 + B_2^2 + B_3^2}\right]}{\sqrt{B_1^2 + B_2^2 + B_3^2}} +$$


$$\frac{(-i B_1 + B_2) S^- (\cos[t B_0] - i \sin[t B_0]) \times \sin\left[t \sqrt{B_1^2 + B_2^2 + B_3^2}\right]}{\sqrt{B_1^2 + B_2^2 + B_3^2}}$$

```

For simplicity, consider the following specific case. We have assumed a certain choice of units.

$B[0] = 0; B[1] = B[3] = 1; B[2] = -1;$

```
In[5]:= op[t_] = Elaborate[U[t]] // ExpToTrig // Garner
Out[5]= Cos[ $\sqrt{3} t$ ] -  $\frac{i S^2 \sin[\sqrt{3} t]}{\sqrt{3}} + \frac{(1-i) S^+ \sin[\sqrt{3} t]}{\sqrt{3}} - \frac{(1+i) S^- \sin[\sqrt{3} t]}{\sqrt{3}}$ 
```

Suppose that the initial state is the eigenstate of the Pauli X operator, here denoted by $s[1]$.

```
In[5]:= v0 = (Ket[] + Ket[S -> 1]) / Sqrt[2];
v0 // LogicalForm
```

$$Out[5]= \frac{|0_S\rangle + |1_S\rangle}{\sqrt{2}}$$

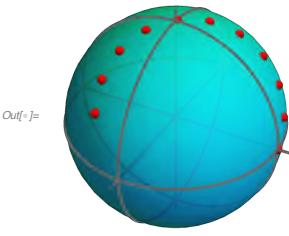
This is the state vector at a later time $t > 0$.

```
In[5]:= vec[t_] = op[t] ** v0;
          vec[t] // LogicalForm

Out[5]= |1s| \left( \frac{\cos[\sqrt{3} t]}{\sqrt{2}} - \frac{\sin[\sqrt{3} t]}{\sqrt{6}} \right) + |0s| \left( \frac{\cos[\sqrt{3} t]}{\sqrt{2}} + \frac{(1-2 i) \sin[\sqrt{3} t]}{\sqrt{6}} \right)
```

This visualizes the evolution of the state under the Hamiltonian on the Bloch sphere.

```
In[5]:= vv = Bead /@ BlochVector /@ Table[vec[t], {t, 0, 1, 0.1}] // Chop;
          BlochSphere[{Red, vv}, ImageSize -> Small]
```



An important point to bear in mind about unitary dynamics is that it does not depend on the history (for a time-independent Hamiltonian). This is reflected in the obvious property of the time-evolution operator,

$$\hat{U}(t + t') = \hat{U}(t)\hat{U}(t') \quad (1.30)$$

for any t, t' . Physically, it implies that the evolution depends only on the duration of time, but not on when it starts or ends. In this respect, it is also natural for a time-evolution operator to have the property $\hat{U}^\dagger(t) = \hat{U}(-t)$.

So far, we have discussed the time evolution of a pure state. What if the system is initially in a mixed state $\hat{\rho}(0)$ for a certain reason? As a statistical mixture of pure states, the mixed state can be expressed as

$$\hat{\rho}(0) = \sum_j |\psi_j(0)\rangle \langle \psi_j(0)| p_j \quad (1.31)$$

with $0 \leq p_j \leq 1$. If the system is closed, then each pure state $|\psi(0)\rangle$ evolves into $\hat{U}(t)|\psi(0)\rangle$. Overall, the state $\hat{\rho}(t)$ at later time t is given by

$$\hat{\rho}(t) = \hat{U}(t)\hat{\rho}(0)\hat{U}^\dagger(t). \quad (1.32)$$

In short, as long as the system remains closed later on, the dynamics is still unitary regardless of whether the system is prepared initially in a pure or mixed state.

Let us consider the same system and Hamiltonian. However, the initial state is now a mixed state.

```
In[6]:= ρθ = vθ ** Dagger[vθ] * 3/4 + Ket[] ** Bra[] ** 1/4 // LogicalForm // Garner
          Out[6]= 
$$\frac{5}{8} |\theta_S\rangle \langle \theta_S| + \frac{3}{8} |\theta_S\rangle \langle 1_S| + \frac{3}{8} |1_S\rangle \langle \theta_S| + \frac{3}{8} |1_S\rangle \langle 1_S|$$

```

Its matrix representation in the logical basis is given by the following.

```
In[7]:= mat = Matrix[ρθ];
          mat // MatrixForm
          Out[7]/MatrixForm= 
$$\begin{pmatrix} \frac{5}{8} & \frac{3}{8} \\ \frac{3}{8} & \frac{3}{8} \end{pmatrix}$$

```

This is the state vector at a later time $t > 0$.

```
In[7]:= Let[Real, t]
rho[t_] = op[t] ** rho0 ** Dagger[op[t]];
rho[t] // LogicalForm
Out[7]= 
$$\frac{5}{8} \cos[\sqrt{3} t]^2 |0_S\rangle\langle 0_S| + \frac{17}{24} |0_S\rangle\langle 0_S| \sin[\sqrt{3} t]^2 + \frac{1}{8} \sqrt{3} |0_S\rangle\langle 0_S| \sin[2\sqrt{3} t] +$$


$$\frac{1}{48} i |0_S\rangle\langle 1_S| \left( (16 + 2i) \sin[\sqrt{3} t]^2 - (6 + 3i) \sqrt{3} \sin[2\sqrt{3} t] \right) +$$


$$\frac{1}{48} |1_S\rangle\langle 0_S| \left( 18 \cos[\sqrt{3} t]^2 - (5 + 2i) \sqrt{3} \sin[2\sqrt{3} t] \right) +$$


$$\frac{1}{48} |0_S\rangle\langle 1_S| \left( 18 \cos[\sqrt{3} t]^2 - (5 - 2i) \sqrt{3} \sin[2\sqrt{3} t] \right) +$$

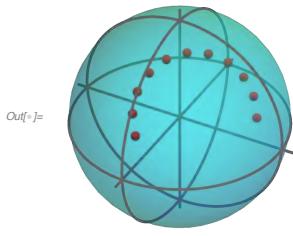

$$\frac{1}{24} |1_S\rangle\langle 1_S| \left( 8 + \cos[2\sqrt{3} t] - 3\sqrt{3} \sin[2\sqrt{3} t] \right) +$$


$$\frac{1}{48} |1_S\rangle\langle 0_S| \left( (-2 - 16i) \sin[\sqrt{3} t]^2 + (3 + 6i) \sqrt{3} \sin[2\sqrt{3} t] \right)$$

```

This visualizes the evolution of the state under the Hamiltonian on the Bloch sphere. Note that the magnitude of the Bloch vectors are preserved by the unitary dynamics.

```
In[8]:= rho = Bead /@ BlochVector /@ Table[rho[t], {t, 0, 1, 0.1}] // Chop;
BlochSphere[{Red, rho}, "Opacity" -> 0.4, ImageSize -> Small]
```



1.2.2 Quantum Noisy Dynamics

When a system interacts with its environment, the dynamics of the system alone cannot be described by the Schrödinger equation any longer, and more importantly, it is not unitary. At a first glance, it may not be surprising as similar effects also occur in classical mechanics. For example, a ball thrown up in the air interacts with various molecules and small particles in the air, and it does not follow Newton's equation of motion. The dynamics becomes dissipative, which is described by additional damping terms in the equation of motion. In effect, the damping terms arise because we are ignoring the small molecules and particles disturbing the ball. However, ignoring the environmental degrees of freedom brings about far more profound effects in quantum theory. It causes not only the dissipation of energy of the system to the environment, but also the effect of so-called *decoherence*, the loss of *quantumness* (Zurek, 1991, 2002). Here we outline the basic formalism of decoherence process, but the details will be discussed in Section 5.1.

Suppose that the system is initially in a state $|\psi\rangle \in \mathcal{H}$ and the environment in $|\epsilon\rangle \in \mathcal{E}$, and thus the initial state of the total system is $|\Psi(0)\rangle = |\psi\rangle \otimes |\epsilon\rangle \in \mathcal{H} \otimes \mathcal{E}$. Then let the system interact with the environment. As a closed system, the

evolution of the total system is governed by a unitary time-evolution operator $\hat{U}_{\text{tot}}(t)$, $|\Psi(t)\rangle = \hat{U}_{\text{tot}}(t) |\psi\rangle \otimes |\epsilon\rangle$, in accordance with Postulate 2. As the system-environment coupling tends to build entanglement, the state $|\Psi(t)\rangle$ cannot be factorized in general. Without a comprehensive knowledge of the environment, the state of the system alone is thus a mixed state as discussed in Section 1.1.2. Specifically, one can get the density matrix representing the mixed state by tracing out the environment information from the total state $|\Psi(t)\rangle$,

$$\hat{\rho}(t) = \text{Tr}_{\mathcal{E}} |\Psi(t)\rangle \langle \Psi(t)| = \text{Tr}_{\mathcal{E}} \hat{U}_{\text{tot}}(t) (|\psi\rangle \langle \psi| \otimes |\epsilon\rangle \langle \epsilon|) \hat{U}_{\text{tot}}^\dagger(t). \quad (1.33)$$

The partial trace over the environment physically corresponds to ignoring the environment degrees of freedom. In general, the dynamics of $\hat{\rho}(t)$ is thus very complicated. But there are two essential points to be noted here: First, the dynamics is not unitary any longer. Second, through the entanglement with the environment, the quantum state of the system loses coherence (i.e., quantumness).

One can describe the non-unitary dynamics of a system coupled to the environment most efficiently in the quantum operation formalism. It is the subject of Chapter 5.

1.3 Measurements on Quantum States

In classical mechanics, just like the state of motion, a physical quantity is described by a simple number (or a set of numbers when it is a vector quantity). As such, measurement is merely about assessing the number and only casts technological questions concerning measuring devices. In quantum mechanics, it is totally different and measurement is deeply intermingled into the quantum theory itself.

1.3.1 Projection Measurements

Postulate 3 A physical quantity is described by an “observable”—a Hermitian operator. Upon *measurement* of an observable \hat{A} , the outcome is one of the eigenvalues a of \hat{A} and determined *probabilistically*. When the system is in the state $|\psi\rangle$ right before the measurement, the probability for a particular outcome a is given by $P_a = |\langle a|\psi\rangle|^2$, where $|a\rangle$ is the eigenstate of \hat{A} belonging to the eigenvalue a . Right after the measurement, the state “collapses” to the eigenstate $|a\rangle$ corresponding to the outcome a .

The postulate points out several striking features that put quantum mechanics in stark contrast with classical mechanics. First, a measurable physical quantity (i.e., an observable) is described by an operator, not by a simple number, acting on the vectors that describe the quantum states of the system (see Postulate 1).

Second, measurement brings about a sudden collapse of the state vector. The unavoidable disturbance by a mere measurement causes obvious obstacles to naively

examining quantum states, but at the same time it opens new opportunities such as secure quantum communication. Further, the collapse is a peculiar type of dynamics of quantum states, an alternative to the one governed by the Schrödinger equation (Postulate 2), which turns out to be useful in measurement-based quantum computing architecture (Section 3.4). Even in common dynamical schemes of quantum computation, measurement is often used to initialize the quantum registers to the logical basis states.

Third, the postulate makes it clear that even if the state vector $|\psi\rangle$ is known, which comprises the complete description of the state according to Postulate 1, one cannot infer the measurement result even in principle, but only the probabilities of possible outcomes. The description is inherently statistical and one need to measure an *ensemble* of identically prepared systems and extract the value of an ensemble in terms of the statistical moments such as the expectation value. Given a state vector $|\psi\rangle$, the expectation value of the operator \hat{A} is obtained using the elementary theory of probability (see also Dirac's Bra-Ket notation in Appendix A.3)

$$\langle \hat{A} \rangle = \sum_q q P_q = \sum_q \langle \psi | q \rangle q \langle q | \psi \rangle = \langle \psi | \hat{A} | \psi \rangle \quad (1.34)$$

A closely related feature of quantum states is reflected in the *no-cloning theorem*: As Zurek (2000) declared, “You can clone a sheep,^{1.3} but not a quantum state,” the *no-cloning theorem* of quantum states dictates that it is impossible to make a copy of an *unknown* quantum state (Wooters & Zurek, 1982). If it were possible, one could produce a bunch of identical copies of the same quantum state. Then, measurements of different incompatible observables (such as conjugate variables) on the copies would reveal the precise information about the quantum state, violating the uncertainty principle. Cloning of (unknown) quantum state would also allow faster-than-light communication. Further, the no-cloning nature of quantum states is one of the key features of quantum mechanics that provide *unconditional* security in quantum communication.

But why is it impossible? Suppose that we have a machine that can do it. The machine would have two quantum registers, the input and target register. The input register is prepared in an (unknown) quantum state $|\psi\rangle$ and the target register in a reference state, say, $|0\rangle$. The overall state of the total system is a tensor-product state $|\psi\rangle \otimes |0\rangle$. Let \hat{U} be the unitary operator corresponding to the operation of copying. By the hypothesis, we expect that

$$\hat{U}(|\psi\rangle \otimes |0\rangle) = |\psi\rangle \otimes |\psi\rangle. \quad (1.35)$$

Let us apply the operation to two orthogonal states $|\alpha\rangle$ and $|\beta\rangle$. We have

$$\hat{U}(|\alpha\rangle \otimes |0\rangle) = |\alpha\rangle \otimes |\alpha\rangle, \quad \hat{U}(|\beta\rangle \otimes |0\rangle) = |\beta\rangle \otimes |\beta\rangle. \quad (1.36)$$

^{1.3}It refers to Dolly, the sheep cloned by a research group (Wilmut *et al.*, 1997).

So far nothing is wrong. Now, let us apply the operation to a linear superposition $|\alpha\rangle a + |\beta\rangle b$ with a and b arbitrary (and unknown) complex numbers. From the linearity of \hat{U} , we have

$$\hat{U}[(|\alpha\rangle a + |\beta\rangle b) \otimes |0\rangle] = \hat{U}(|\alpha\rangle \otimes |0\rangle a + |\beta\rangle \otimes |0\rangle b) = |\alpha\rangle \otimes |\alpha\rangle a + |\beta\rangle \otimes |\beta\rangle b. \quad (1.37)$$

Clearly, the result is different from what we expect,

$$(|\alpha\rangle a + |\beta\rangle b) \otimes (|\alpha\rangle a + |\beta\rangle b), \quad (1.38)$$

unless either a or b vanishes. Therefore, we conclude that copying a quantum state is impossible. The crucial point in the proof is the linearity of quantum mechanics.

In the above statement of the postulate, the system was assumed to be initially in a pure state. It is naturally extended to the case of mixed state: When a system is in a mixed state $\hat{\rho}$ right before the measurement, the probability is given by $P_a = \langle a| \hat{\rho}|a\rangle$ and the state right after the measurement becomes $|a\rangle \langle a|$. By virtue of the elementary theory of probability, we observe again that $0 \leq \hat{\rho} \leq 1$ and $\text{Tr } \hat{\rho} = 1$ as we have already seen in Section 1.1.2. Further, the expectation value of the observable is given by

$$\langle \hat{A} \rangle = \sum_a a P_a = \sum_a a \langle a| \hat{\rho}|a\rangle = \text{Tr } \hat{A} \hat{\rho}. \quad (1.39)$$

Von Neumann envisioned an idealistic procedure to realize a projection measurement on physical systems. For this reason, projection measurement is also called as von Neumann measurement. The *von Neumann scheme* founded the quantum theory of measurement and has inspired various methods to push the measurement precision to the limit intrinsically put by quantum mechanics. The von Neumann scheme of projection measurement can be directly implemented in a quantum circuit model (see Section 4.4). Here briefly summarize the procedure: Suppose that we want to measure a quantity described by the Hermitian operator \hat{A} . Let $|a\rangle$ be the eigenvectors with eigenvalues a of \hat{A} so that \hat{A} has the spectral decomposition (see Appendix A.3)

$$\hat{A} = \sum_a |a\rangle a \langle a|. \quad (1.40)$$

The system is in a state $|\psi\rangle$, which is expanded as

$$|\psi\rangle = \sum_a |a\rangle \psi_a \quad (1.41)$$

in the eigenbasis of \hat{A} . Typically, it is supposed that the distribution $P_a := |\psi_a|^2$ is sharply peaked around a certain unknown eigenstate $|a_*\rangle$ and the measurement is supposed to reveal a_* .

We choose a measuring device the “position” \hat{X} of which can be directly observed, and prepare it in an approximate eigenstate $|\xi\rangle$ of \hat{X} . In terms of the eigenstates $|x\rangle$ with eigenvalue x of \hat{X} , $|\xi\rangle$ is expanded as

$$|\xi\rangle = \int dx |x\rangle \xi(x) \quad (1.42)$$

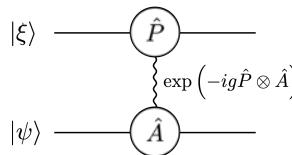
with $\xi(x)$ peaked sharply around a certain fixed point x_0 . It will get clear below that the sharper the distribution $|\xi(x)|^2$ in the position is, the more accurate the measurement result gets. At this stage, the overall state of the total system composed of the system in question and the measurement device is given by $|\psi\rangle \otimes |\xi\rangle$. Now couple the observable \hat{A} in question to the “momentum” \hat{P} (not \hat{X} itself) of the measurement device. Note that \hat{P} and \hat{X} are canonical conjugates of each other, $[\hat{X}, \hat{P}] = i$ (we have chosen a unit system such that $\hbar = 1$). The coupling is described by the interaction Hamiltonian

$$\hat{H}_{\text{int}} = J\hat{P} \otimes \hat{A}, \quad (1.43)$$

where J is the coupling strength. One has to let the system and the measurement device interact for sufficiently long time τ such that the position of the measurement device gets distinguished from the initial position. Let $g := J\tau$ be the dimensionless coupling constant. Due to the coupling, the total system composed of the system in question and the measurement device evolves in time, which is described by the unitary operator

$$\hat{U}_{\text{int}} = \exp(-ig\hat{P} \otimes \hat{A}). \quad (1.44)$$

This situation is depicted diagrammatically as following



It is instructive to expand the interaction unitary operator \hat{U}_{int} using the spectral decomposition [see Eq. (A.42)] of the observable \hat{A} as

$$\hat{U}_{\text{int}} = \sum_a e^{-iga\hat{P}} \otimes |a\rangle \langle a| \quad (1.45)$$

The operator $\hat{T}_a := e^{-iga\hat{P}}$ is a translation operator with the amount of translation depending on the eigenvalue a of \hat{A} , and to make the physical interpretation clearer, we rewrite the interaction unitary operator into

$$\hat{U}_{\text{int}} = \sum_a \hat{T}_a \otimes |a\rangle \langle a|. \quad (1.46)$$

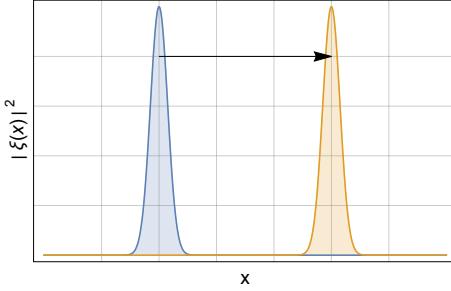


Figure 1.3: The change of the wave function $\xi(x)$ of the measurement device from the initial state (blue) to the final state (orange) after it interacts with the system. Illustrated is the case where the system is in a definite eigenstate $|a\rangle$ of the observable \hat{A} to be measured. The amount of translation is given by ga , the coupling constant times the eigenvalue associated with $|a\rangle$.

This picture is illustrated in the following schematic diagram



After the interaction, the state vector of the total system becomes

$$\hat{U}_{\text{int}} |\Psi\rangle = \sum_a (\hat{T}_a |\xi\rangle) \otimes (|a\rangle \psi_a) = \int dx |x\rangle \otimes \left(\sum_a |a\rangle \xi(x - ga) \psi_a \right). \quad (1.48)$$

In general, the final state is an entangled state. The probability to register x out of the measurement on the measurement device is given by

$$P(x) = \sum_a |\xi(x - ga)|^2 |\psi_a|^2. \quad (1.49)$$

When the system starts from a definite eigenstate $|a\rangle$ of \hat{A} , the system and the measurement device remain unentangled after the system-measurement device interaction, with the probability distribution determined solely by the given eigenvalue a

$$P(x) = |\xi(x - ga)|^2. \quad (1.50)$$

Manifestly, the measurement accuracy is best in this case. As illustrated in Fig. 1.3, one can extract the eigenvalue a of the observable from the amount of translation ga of the final wave function relative to the initial wave function assuming that the coupling strength g is known (it can be calibrated separately).

Figure 1.3 suggests that the sharper the initial wave function of the measurement device is, the more precise the extracted value of a becomes. Roughly speaking, this

is indeed one direction of efforts to improve the precision of measurement (Caves, 1981). An obstacle to precision measurement that is not directly apparent in Fig. 1.3 is the inherent statistical nature of quantum mechanics. As the probability distribution has a finite width, any measurement in quantum mechanics should be repeated many times. The statistical error decreases as $1/\sqrt{N}$ with the number N of repeated measurements. This is called the *standard quantum limit*. Interestingly, if one prepares a set of measurement devices in a proper entangled state, one can improve the statistical error to $1/N$ (Giovannetti *et al.*, 2006). This enhanced accuracy due to quantum entanglement is called the *Heisenberg limit*, and it puts the ultimate limit on the measurement precision that quantum mechanics allows.

1.3.2 Generalized Measurements

We have already introduced a Postulate 3 concerning measurement, and it is usually the form of the measurement postulate discussed in most textbooks on quantum mechanics. In quantum information context, however, it is more convenient to generalize it for various reasons to be discussed shortly.

Postulate 3' A generalized measurement is described by a set of *measurement operators* \hat{M}_m corresponding to measurement outcomes m that satisfy the completeness relation $\sum_m \hat{M}_m^\dagger \hat{M}_m = \hat{I}$. Suppose that right before the measurement, the system is in the quantum state $\hat{\rho}$. Then the probability P_m for the outcome m is given by $P_m = \text{Tr } \hat{M}_m \hat{\rho} \hat{M}_m^\dagger$, and the state right after measurement becomes $P_m^{-1} \hat{M}_m \hat{\rho} \hat{M}_m^\dagger$.

Projective measurement discussed previously is a special case of generalized measurement. Suppose that we measure an observable \hat{A} . In this case, the available measurement outcomes are the eigenvalues a of \hat{A} . The measurement operator corresponding to the outcome a is the projection operator $|a\rangle\langle a|$ into the eigenstate $|a\rangle$ belonging to the eigenvalue a .

In general, the physical meaning of the measurement operators \hat{M}_m is not immediately clear until the specifics of the measurement in question are given. It is even more obscure how to set up a physical device for the measurement described by the given set of operators \hat{M}_m . Nevertheless, one can show that generalized measurements are equivalent to projective measurements (extended to a larger system and supplemented with additional unitary operations). In many applications, however, generalized measurements are more convenient because they are less restrictive.

For example, consider a fixed set of *orthogonal* quantum states $|\psi_1\rangle, \dots, |\psi_n\rangle$ (not necessarily spanning the Hilbert space). Suppose that a state is drawn from the set. The task is to tell which of the n state it is. We proceed by defining n operators $\hat{M}_j := |\psi_j\rangle\langle\psi_j|$ for $j = 1, \dots, n$ and an additional operator $\hat{M}_0 := \hat{I} - \sum_j \hat{M}_j$. They satisfy the completeness relation $\sum_{j=0}^n \hat{M}_j^\dagger \hat{M}_j = \hat{I}$ and describe a measurement.

Now when the particular state $|\psi_k\rangle$ is taken, the probability P_k for the measurement outcome k is unity because $P_k = \langle\psi_k| \hat{M}_k |\psi_k\rangle = 1$.

Postulate 3 and 3' describe not only the probability for a particular measurement outcome but also the state corresponding to the outcome after the measurement. In many experiments, however, the system is measured only once, and the state after the measurement is irrelevant. Of primary concern is the probabilities. Often, it is even impossible to assign a post-measurement state to the system. For example, in an experiment to measure the position of photon, the photon is destroyed at the photo-screen, and it is meaningless to ask about the wave function of the photon after measurement.

As long as one is focused on the probabilities, the description in Postulate 3' can be drastically simplified: Since the trace is invariant under cyclic permutation of the operators, the probability for the outcome m is given by

$$P_m = \text{Tr } \hat{M}_m \hat{\rho} \hat{M}_m^\dagger = \text{Tr } \hat{M}_m^\dagger \hat{M}_m \hat{\rho}. \quad (1.51)$$

Therefore, as long as the probabilities are concerned, all we need are the operators $\hat{E}_m := \hat{M}_m^\dagger \hat{M}_m$ rather than the measurement operators \hat{M}_m . Note that \hat{M}_m may contain far more details that cannot be extracted from \hat{E}_m . Infinitely many different measurement operators can lead to the same \hat{E}_m . Nevertheless, it is possible to discard certain details in \hat{M}_m for more compact operators \hat{E}_m because we do not care about the post-measurement state. By construction, \hat{E}_m are positive semi-definite operators and satisfy $\sum_m \hat{E}_m = \hat{I}$. The set $\{\hat{E}_m\}$ of such operators is called a *positive operator-valued measure* or more often a POVM. The individual operators \hat{E}_m are called the *POVM elements*. For many experiments, the POVM formalism allows far more efficient description of the measurement.

As a non-trivial example of the application of the POVM formalism, let us consider the problem of quantum state discrimination of *non-orthogonal* quantum states (Bergou *et al.*, 2004; Chefles, 2004). To be specific, suppose that we are given either of the two states

$$|v\rangle = |0\rangle, \quad |w\rangle = \frac{|0\rangle + |1\rangle}{\sqrt{2}}. \quad (1.52)$$

We want to figure out which it is. Apparently, these two states are not orthogonal, $\langle v|w\rangle \neq 0$, and it is impossible to discriminate them with certainty. Instead, the task is to perform a measurement that tells which of the two, but never misidentifies a wrong state. Consider a POVM including the following three elements

$$\hat{E}_1 = \frac{1}{\sqrt{1 + \langle v|w\rangle}} |1\rangle \langle 1|, \quad (1.53a)$$

$$\hat{E}_2 = \frac{1}{\sqrt{1 + \langle v|w\rangle}} \left(\frac{|0\rangle - |1\rangle}{\sqrt{2}} \right) \left(\frac{\langle 0| - \langle 1|}{\sqrt{2}} \right) \quad (1.53b)$$

$$\hat{E}_3 = \hat{I} - \hat{E}_1 - \hat{E}_2 \quad (1.53c)$$

It is straightforward to check that they form a POVM. Suppose that the state $|v\rangle$ is given. As it is orthogonal to $|1\rangle$, the measurement outcome $m = 1$ —corresponding to the POVM element \hat{E}_1 —never occurs. It means that once $m = 2$ —corresponding to \hat{E}_2 —occurs, the state must be $|w\rangle$. For the same reason, the result $m = 1$ implies that the state is definitely $|v\rangle$. Of course, it is possible to get the outcome $m = 3$, in which case the measurement tells nothing. However, the important point is that there is no chance to make a mistake.

Problems

- 1.1. Consider a set of *arbitrary* state vectors $\{|\psi_\mu\rangle : \mu = 0, 1, \dots, n - 1\}$ in a vector space. Note that the vectors are not normalized nor orthogonal to each other. Let \hat{A} be a linear operator defined by

$$\hat{A} = \sum_{\mu=0}^{n-1} |\psi_\mu\rangle \langle \psi_\mu| . \quad (1.54)$$

Show that the eigenvalues of \hat{A} are all non-negative.

- 1.2. Let $\hat{\rho}$ be a density operator. Prove the following statements.

- (a) $\text{Tr } \hat{\rho} = 1$.
- (b) $\hat{\rho}$ is positive (Definition A.12).
- (c) $0 \leq \lambda \leq 1$ for any eigenvalue λ of $\hat{\rho}$.

- 1.3. Show that a density operator $\hat{\rho}$ is a pure state if and only if $\text{Tr } \hat{\rho}^2 = 1$.

- 1.4. Consider a two-level quantum system. Let $|0\rangle$ and $|1\rangle$ be the logical basis of the Hilbert space associated with the system. Let \hat{S}^μ ($\mu = 0, x, y, z$) be the Pauli operator, defined by

$$\hat{S}^x |0\rangle = |1\rangle , \quad \hat{S}^y |0\rangle = +i |1\rangle , \quad \hat{S}^z |0\rangle = + |0\rangle , \quad (1.55a)$$

$$\hat{S}^x |1\rangle = |0\rangle , \quad \hat{S}^y |1\rangle = -i |0\rangle , \quad \hat{S}^z |1\rangle = - |1\rangle , \quad (1.55b)$$

and $\hat{S}^0 = \hat{I}$.

- (a) Find the matrix representations of \hat{S}^μ ($\mu = 0, x, y, z$) in the logical basis.
- (b) Find the matrix representations of \hat{S}^μ in the new basis

$$|\pm\rangle := \frac{|0\rangle \pm |1\rangle}{\sqrt{2}} . \quad (1.56)$$

- 1.5. Consider a spin 1/2 in an external magnetic field. We describe the states of the spin with the vector space spanned by the logical basis states $|0\rangle \equiv |\uparrow\rangle$ and $|1\rangle \equiv |\downarrow\rangle$. In terms of the Pauli operators in Eq. (1.55), the Hamiltonian is given by

$$\hat{H} = B_x \hat{S}^x + B_y \hat{S}^y + B_z \hat{S}^z, \quad (1.57)$$

where B_μ ($\mu = x, y, z$) are the parameters proportional to the external magnetic field—in this expression, they have the same dimension as the energy. Suppose that

$$B_x = B_0 \sin \theta \cos \phi, \quad B_y = B_0 \sin \theta \sin \phi, \quad B_z = B_0 \cos \theta \quad (1.58)$$

with $B_0 > 0$ and $\theta, \phi \in \mathbb{R}$.

- (a) Find the eigenvalues and corresponding eigenstates of \hat{H} .

Hint: Direct evaluation of the Hamiltonian in Eq. (1.57) is straightforward. Another method is to use the commutation relations of the Pauli operators.

- (b) Display the two eigenstates on the Bloch sphere.

- 1.6. Let $|\psi\rangle$ be a vector in a two-dimensional vector space. Show that the Bloch vector $\mathbf{b} := (\langle \hat{S}^x \rangle, \langle \hat{S}^y \rangle, \langle \hat{S}^z \rangle) \in \mathbb{R}^3$ has the magnitude of unity, $|\mathbf{b}| = 1$.

- 1.7. Let $\hat{\rho}$ be a density operator on a two-dimensional vector space. It can be written as

$$\hat{\rho} = \frac{1}{2} \left(\hat{S}^0 + x \hat{S}^x + y \hat{S}^y + z \hat{S}^z \right), \quad (1.59)$$

where \hat{S}^μ are the Pauli operators—see Eq. (1.56)—and $x, y, z \in \mathbb{R}$.

- (a) Show that the Bloch vector defined by $\mathbf{b} := (\langle \hat{S}^x \rangle, \langle \hat{S}^y \rangle, \langle \hat{S}^z \rangle)$ is just given by $\mathbf{b} = (x, y, z)$.
- (b) Show that $|\mathbf{b}| \leq 1$.
- (c) Show that $\hat{\rho}$ is a pure state if and only if $|\mathbf{b}| = 1$.

Hint: See Problem 1.5.

- 1.8. Consider a two-qubit system, and suppose that it is in the state

$$|\psi\rangle = \frac{|0\rangle \otimes |0\rangle + |0\rangle \otimes |1\rangle + |1\rangle \otimes |0\rangle}{\sqrt{3}}. \quad (1.60)$$

- (a) What is the probability p_0 to find the first qubit in $|0\rangle$ (regardless of the second qubit)? Similarly, what is the probability p_2 to find the first qubit in the state $|1\rangle$?
- (b) What is the probability p'_0 to find the first qubit in $(|0\rangle + |1\rangle)/\sqrt{2}$? What about the probability p'_1 to find the first qubit in $|0\rangle$?

- (c) Suppose that you do not have an access to the second qubit, and that you want to construct the density operator $\hat{\rho}$ for the first qubit corresponding to the physical situation. Which of the two data sets from (1.8a) and (1.8b) would you use for the construction of $\hat{\rho}$? That is, if the resulting density operators are different, which one is correct and why?
- (d) Calculate the Bloch vector \mathbf{b} corresponding to $\hat{\rho}$ in (b), and display it in the Bloch sphere.

1.9. Consider a system of two two-level subsystems. The logical basis and the Pauli operators for each subsystem are defined as in Problem 1.4.

- (a) Find the matrix representations of the following operator

$$\hat{H} = \hat{S}^x \otimes \hat{S}^x + \hat{S}^y \otimes \hat{S}^y + \hat{S}^z \otimes \hat{S}^z \quad (1.61)$$

in the standard tensor-product basis $\{|i\rangle \otimes |j\rangle : i, j = 0, 1\}$.

- (b) Find all eigenvalues and eigenvectors of \hat{H} .

- (c) Calculate the following unitary operator

$$\hat{U} = \exp(-it\hat{H}), \quad t \in \mathbb{R}, \quad (1.62)$$

which corresponds physically to the time-evolution operator, and express it in terms of either $\hat{S}^\mu \otimes \hat{S}^\nu$ or the dyadic products, $|i\rangle \langle j| \otimes |k\rangle \langle l|$ ($i, j, k, l = 0, 1$).

- (d) Evaluate the state

$$|\psi(t)\rangle := \hat{U}(t) |+\rangle \otimes |-\rangle, \quad (1.63)$$

where $|\pm\rangle$ are defined in Eq. (1.56).

- (e) Evaluate the expectation values

$$S_1^\mu(t) := \langle \psi(t) | \hat{S}^\mu \otimes \hat{S}^0 | \psi(t) \rangle, \quad (1.64)$$

for $\mu = x, y, z$, and plot them as functions of t .

- (f) Evaluate the expectation values

$$S^\mu(t) := \langle \psi(t) | \left(\hat{S}^\mu \otimes \hat{S}^0 + \hat{S}^0 \otimes \hat{S}^\mu \right) | \psi(t) \rangle, \quad (1.65)$$

for $\mu = x, y, z$, and plot them as functions of t .

1.10. Consider a two-qubit system in the following state

$$|\psi\rangle = \frac{1}{2} \sum_{x=0}^3 |x\rangle. \quad (1.66)$$

Here we have used a short-hand notation $|x\rangle := |x_1\rangle \otimes |x_2\rangle$, where x_j ($j = 1, 2$) are the binary digits of x , $x = (x_1 x_2)_2$. For example, $|2\rangle = |1\rangle \otimes |0\rangle$. Suppose that we measure an observable

$$\hat{H} = \sum_{\mu \in \{x,y\}} \hat{S}^\mu \otimes \hat{S}^\mu, \quad (1.67)$$

where \hat{S}^μ ($\mu = x, y, z$) are the Pauli operators.

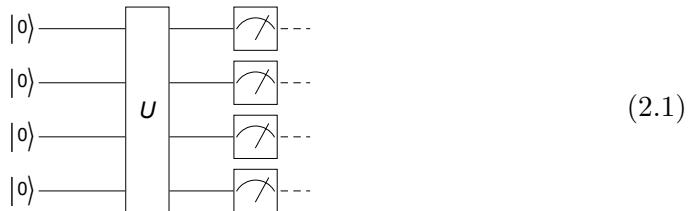
- (a) Find all possible values of the outcome of the measurement according to the postulates of quantum mechanics.
- (b) For each of the values found in (a), find the probability to actually observe the value as the measurement outcome. Plot the probabilities versus the possible values, say, in a bar chart to compare them directly.
- (c) For each of the values found in (a), find the post-measurement state when the value actually is registered as the measurement outcome.

Chapter 2

Quantum Computation: Overview

- August 4, 2021 (v1.18)

In the simplest physical terms, quantum computation is an implementation of an arbitrary unitary operation on a finite collection of *quantum bits* or *qubits* for short—a two-level quantum system. It is typically depicted in a *quantum circuit diagram* as follows:



Each qubit is associated with a line that indicates the time evolution of the state specified on the left. The time flows from the left to the right. The *quantum logic gate operations* (or *gates* for short) on single or multiple qubits are denoted by a rectangular box often with labels indicating the types of the gates. Measurements are denoted by square boxes with needles. After a measurement, the time-evolution is represented by dashed lines to remind that the information is classical—no superposition.

The input state is prepared in one of the states in the logical basis, typically $|0\rangle \otimes \cdots \otimes |0\rangle$. After the unitary operation, the resulting state is measured in the logical basis, and the readouts are supposed to be the result of the computation.

In order for a quantum computer to be programmable, it is required to implement a given unitary operator \hat{U} in a combination of other more elementary unitary operators

$$\hat{U} = \hat{U}_1 \hat{U}_2 \cdots \hat{U}_L, \quad (2.2)$$

where each \hat{U}_j is chosen from a small fixed set of elementary gate operations. The latter operations are called the *elementary quantum logic gates* for the quantum computer. In this chapter, we will examine widely used choices for elementary gates, and illustrate how a set of elementary gates achieve the arbitrary unitary operation, the so-called *universal quantum computation*.

Throughout the chapter, we denote by \mathcal{S} the Hilbert space associated with a single qubit. The Hilbert space of an n -qubit system is given by $\mathcal{S}^{\otimes n}$, a tensor-product space of multiple \mathcal{S} . Each element $|x\rangle$ in the logical basis of $\mathcal{S}^{\otimes n}$ will be labeled by an integer $x = 0, 1, \dots, 2^n - 1$, which should be understood to enumerate the tensor product form

$$|x\rangle \equiv |x_1 x_2 \cdots x_n\rangle := |x_1\rangle \otimes |x_2\rangle \otimes \cdots \otimes |x_n\rangle \quad (2.3)$$

in terms of the binary digits x_j ($j = 1, 2, \dots, n$) of x , that is, $x \equiv (x_1 x_2 \cdots x_n)_2$.

2.1 Single-Qubit Gates

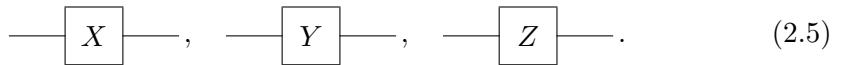
Unitary operators on the two-dimensional vector space \mathcal{S} associated with a singel qubit form the *unitary group* $U(2)$. In the standard logical basis, they are represented by 2×2 unitary matrices. We first take a look at some special examples and discuss the general properties of the single-qubit unitary operations.

2.1.1 Pauli Gates

The Pauli gate opertaions (or Pauli operators for short) are defined by the corresponding Pauli matrices

$$\hat{X} \doteq \begin{bmatrix} 0 & 1 \\ 1 & 0 \end{bmatrix}, \quad \hat{Y} \doteq \begin{bmatrix} 0 & -i \\ i & 0 \end{bmatrix}, \quad \hat{Z} \doteq \begin{bmatrix} 1 & 0 \\ 0 & -1 \end{bmatrix}. \quad (2.4)$$

They form the most elementary single-qubit gate operations and are frequently used in many quantum algorithms. In this book, the Pauli gates will be sometimes denoted by \hat{X} , \hat{Y} , and \hat{Z} , and othertimes by \hat{S}^x , \hat{S}^y , and \hat{S}^z , depending on the context. In the quantum circuit model, they are typically depicted by the circuit elements



$$\text{---} \boxed{X} \text{---}, \quad \text{---} \boxed{Y} \text{---}, \quad \text{---} \boxed{Z} \text{---}. \quad (2.5)$$

Pauli \hat{X} maps the logical basis states as

$$\hat{X} : |0\rangle \mapsto |1\rangle, \quad |1\rangle \mapsto |0\rangle, \quad (2.6)$$

and is similar to the classical logic gate NOT. It is also customary to write Pauli \hat{X} in the bra-ket notation as

$$\hat{X} = |1\rangle\langle 0| + |0\rangle\langle 1|. \quad (2.7)$$

It is important to remember that like any other quantum gate operations, it can take a linear superposition as input and transform the two logical basis states “simultaneously”,

$$\hat{X}(|0\rangle c_0 + |1\rangle c_1) = |1\rangle c_0 + |0\rangle c_1, \quad (2.8)$$

which is not possible with the classical counterpart NOT.

The Pauli X gate is represented by `s[...,1]`.

`In[1]:= S[1]`

`Out[1]:= S^x`

It corresponds to the Pauli X matrix.

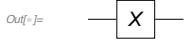
`In[2]:= Matrix[S[1]] // MatrixForm`

`Out[2]:= MatrixForm=`

$$\begin{pmatrix} 0 & 1 \\ 1 & 0 \end{pmatrix}$$

In the quantum circuit model, it is denoted by the following quantum circuit element.

`In[3]:= QuantumCircuit[S[1]]`



Operating on the logical basis states, it flips the states and is similar to the classical logical gate NOT.

`In[4]:= bs = Basis[S];`

`out = S[1] ** bs;`

`Thread[bs \[Rule] out] // LogicalForm // TableForm`

`Out[4]:= TableForm=`

$$|0_S\rangle \rightarrow |1_S\rangle$$

$$|1_S\rangle \rightarrow |0_S\rangle$$

Operating on a superposition state, it flips the state “simultaneously”.

`In[5]:= in = Ket[] \times c[0] + Ket[S \[Rule] 1] \times c[1];`

`in // LogicalForm`

`out = S[1] ** in;`

`out // LogicalForm`

`Out[5]:= TableForm=`

$$c_0 |0_S\rangle + c_1 |1_S\rangle$$

`Out[5]:= TableForm=`

$$c_1 |0_S\rangle + c_0 |1_S\rangle$$

Operating on the logical basis states, Pauli \hat{Z} only changes the relative phase of $|1\rangle$,

$$\hat{Z} : |0\rangle \mapsto |0\rangle, |1\rangle \mapsto -|1\rangle, \quad (2.9)$$

and hence in the bra-ket notation, it reads as

$$\hat{Z} = |0\rangle\langle 0| - |1\rangle\langle 1|. \quad (2.10)$$

The phase change is meaningless on classical bits, but it makes a significant difference on a superposition as illustrated in the following example

$$\hat{Z}(|0\rangle c_0 + |1\rangle c_1) = |0\rangle c_0 - |1\rangle c_1. \quad (2.11)$$

The Pauli Z gate is represented by `s[...,3]`.

```
In[1]:= S[3]
Out[1]= Sz
```

It corresponds to the Pauli Z matrix.

```
In[2]:= Matrix[S[3]] // MatrixForm
Out[2]//MatrixForm=

$$\begin{pmatrix} 1 & 0 \\ 0 & -1 \end{pmatrix}$$

```

In the quantum circuit model, it is denoted by the following quantum circuit element.

```
In[3]:= QuantumCircuit[S[3]]
Out[3]=
```



Operating on the logical basis states, it “flips the phase”, that is, it changes the phase factor to -1 of the logical basis state $|1\rangle$.

```
In[4]:= bs = Basis[S];
out = S[3] ** bs;
Thread[bs → out] // LogicalForm // TableForm
Out[4]//TableForm=

$$\begin{aligned} |0_S\rangle &\rightarrow |0_S\rangle \\ |1_S\rangle &\rightarrow -|1_S\rangle \end{aligned}$$

```

Here is an example how the Pauli Z gate acts on a superposition state.

```
In[5]:= in = Ket[] × c[0] + Ket[S → 1] × c[1];
in // LogicalForm
out = S[3] ** in;
out // LogicalForm
Out[5]= c0 |0S1 |1S0 |0S1 |1S

```

Pauli \hat{Y} combines the bit-flip feature of \hat{X} and the phase-flip feature of \hat{Z} to get

$$|0\rangle \mapsto i|1\rangle, \quad |1\rangle \mapsto -i|0\rangle. \quad (2.12)$$

This can also be seen in the operator identity, $\hat{Y} = i\hat{X}\hat{Z}$. In the bra-ket notation, it reads as

$$\hat{Y} = i|1\rangle\langle 0| - i|0\rangle\langle 1|. \quad (2.13)$$

The Pauli Y gate is represented by `s[...,2]`.

```
In[1]:= S[2]
Out[1]= Sy
```

It corresponds to the Pauli Y matrix.

```
In[2]:= Matrix[S[2]] // MatrixForm
Out[2]//MatrixForm=

$$\begin{pmatrix} 0 & -i \\ i & 0 \end{pmatrix}$$

```

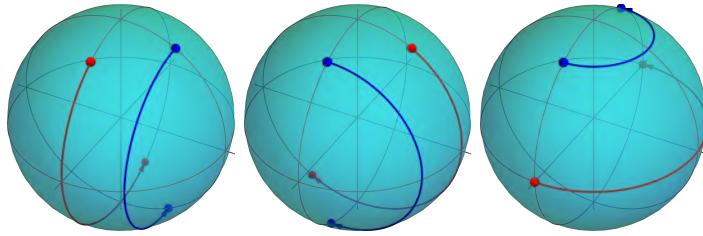


Figure 2.1: Illustration of the actions of Pauli gates as rotations by angle π . From the left the actions of Pauli \hat{X} , \hat{Y} , \hat{Z} .

In the quantum circuit model, it is denoted by the following quantum circuit element.

```
In[1]:= QuantumCircuit[S[2]]
          ──[Y]─
```

Pauli Y is a combination of the bit-flip (Pauli X) and phase-flip (Pauli Z) operation.

```
In[2]:= qc = QuantumCircuit[S[1], S[3]];
op = ExpressionFor[qc]
Out[2]= ──[X]─[Z]─
```

```
Out[2]=  $i \cdot S^y$ 
```

This shows more explicitly how Pauli Y “flips” both the bit and phase of the logical basis states.

```
In[3]:= bs = Basis[S];
out = S[2] ** bs;
Thread[bs → out] // LogicalForm // TableForm
Out[3]/TableForm=

$$\begin{aligned} |0_S\rangle &\rightarrow i |1_S\rangle \\ |1_S\rangle &\rightarrow -i |0_S\rangle \end{aligned}$$

```

Here is an example how the Pauli Y gate acts on a superposition state.

```
In[4]:= in = Ket[]  $\times$  c[0] + Ket[S → 1]  $\times$  c[1];
in // LogicalForm
out = S[2] ** in;
out // LogicalForm
Out[4]=  $c_0 |0_S\rangle + c_1 |1_S\rangle$ 
Out[5]=  $-i c_1 |0_S\rangle + i c_0 |1_S\rangle$ 
```

The Pauli gates can also be regarded as rotations by π around the x -, y -, and z -axis, respectively, in the Bloch sphere as illustrated in Fig. 2.1 and demonstrated in the following:

The Pauli gates also correspond, up to a global phase factor ($-i$), to rotations by the corresponding axes by angle π . Here `Rotation[φ, S[..., μ]]` represents the rotation operator around the μ -axis by angle $φ$.

```
In[=]:= Rotation[Pi, S[1]]
          Rotation[Pi, S[1]] // Elaborate
Out[=]= Rotation[\pi, Sx]
Out[=]= - \mathbb{I} Sx

In[=]:= Rotation[Pi, S[2]] // Elaborate
Out[=]= - \mathbb{I} Sy

In[=]:= Rotation[Pi, S[3]] // Elaborate
Out[=]= - \mathbb{I} Sz
```

Here we are mainly focusing on their roles as unitary operators. However, the Pauli operators play another important role as an orthogonal *basis vectors* of the vector space of all linear operators on a two-dimensional vector space—see Section 2.1.3 and Appendix B.1.

2.1.2 Hadamard Gate

The Hadamard gate is one of the most frequently used elementary gates in many quantum algorithms. The Hadamard gate \hat{H} is defined by the mapping:

$$|0\rangle \mapsto \frac{|0\rangle + |1\rangle}{\sqrt{2}}, \quad |1\rangle \mapsto \frac{|0\rangle - |1\rangle}{\sqrt{2}}. \quad (2.14)$$

That is, it constructs linear superpoisitions of the two logical basis states. It is this feature that makes the Hadamard gate so useful, and is exploited in a wide range of quantum algorithms. In the logical basis, it is represented by the 2×2 Hadamard matrix

$$\hat{H} \doteq \frac{1}{\sqrt{2}} \begin{bmatrix} 1 & 1 \\ 1 & -1 \end{bmatrix} \quad (2.15)$$

In the quantum circuit model, the Hadamard gate is depicted by an element with label “H”



Note that the output states in (2.14) corresponds to the eigenstates of the Pauli X operator. One can thus regard the Hadamard gate as a basis transformation from the logical basis to the eigenbasis of the Pauli X gate.

The Hadamard gate is represented by $S[..., 6]$.

```
In[=]:= S[1, 6]
Out[=]= SH1
```

Let us consider all the logical basis states.

```
In[=]:= bs = Basis[S[1]];
          bs // LogicalForm
Out[=]= {|\theta_{S_1}\rangle, |1_{S_1}\rangle}
```

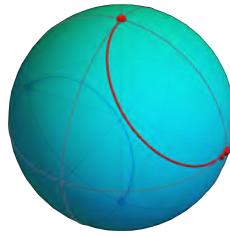


Figure 2.2: Illustration of the action of the Hadamard gate on the Bloch sphere. The Hadamard gate corresponds to a rotation around the axis $(1, 0, 1)$ in the xz -plane by angle π .

Operating the Hadamard gate on them gives the two superposition states.

```
In[1]:= out = S[1, 6] ** bs;
          out // LogicalForm
Out[1]= { |0_{S_1}\rangle + |1_{S_1}\rangle / \sqrt{2}, |0_{S_1}\rangle - |1_{S_1}\rangle / \sqrt{2} }
```

In the quantum circuit model, it is denoted by the following circuit element.

```
In[2]:= QuantumCircuit[S[1, 6]]
Out[2]= ─── [H] ───
```

It is also insightful to note that it can be regarded (up to a global phase factor) as a rotation by angle π around the axis $(1, 0, 1)$ on the Bloch sphere. This can be seen from the following:

$$\hat{H} = \frac{1}{\sqrt{2}} (\hat{X} + \hat{Z}) = i \exp \left[-i \frac{\pi}{2} (\hat{X} + \hat{Z}) \right] \quad (2.17)$$

This is illustrated in Fig. 2.2.

Geometrically, the Hadamard gate can be regarded as a rotation around the axis $(1, 0, 1)$ in the Pauli space by angle π .

```
In[3]:= op = I.Rotation[\pi, S, {1, 0, 1}] // Elaborate
          mat = Matrix[op];
          mat // MatrixForm
Out[3]= S^x / \sqrt{2} + S^z / \sqrt{2}
Out[3]//MatrixForm=
\begin{pmatrix} \frac{1}{\sqrt{2}} & \frac{1}{\sqrt{2}} \\ \frac{1}{\sqrt{2}} & -\frac{1}{\sqrt{2}} \end{pmatrix}
```

An obvious but very useful feature is that it makes a linear superposition of *all* states in the logical basis: Consider a *quantum register* consisting of n qubits.

When applied to each qubit in $|0\rangle$, it generates a linear superposition of all states in the logical basis

$$\hat{H}^{\otimes n} |0\rangle^{\otimes n} = \frac{|0\rangle + |1\rangle}{\sqrt{2}} \otimes \cdots \otimes \frac{|0\rangle + |1\rangle}{\sqrt{2}} = \frac{1}{2^{n/2}} \sum_{x=0}^{2^n-1} |x\rangle , \quad (2.18)$$

where $|x\rangle := |x_1\rangle \otimes |x_2\rangle \otimes \cdots \otimes |x_n\rangle$ for an integer x represented by $x = (x_1 x_2 \dots x_n)_2$ in the binary digits. More generally, for an arbitrary state $|y\rangle$ in the logical basis,

$$\hat{H}^{\otimes n} |y\rangle = \frac{1}{2^{n/2}} \sum_{x=0}^{2^n-1} |x\rangle (-1)^{x \cdot y} , \quad (2.19)$$

where we have used a short-hand notation

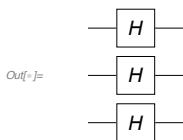
$$x \cdot y := x_1 y_1 + \cdots + x_n y_n \pmod{2} . \quad (2.20)$$

Suppose the Hadamard gates are applied to three qubits.

```
In[7]:= op = HoldForm@Multiply[S[1, 6], S[2, 6], S[3, 6]]
Out[7]= SH1 SH2 SH3
```

This shows the overall operation in the quantum circuit model.

```
In[8]:= qc = QuantumCircuit[S[{1, 2, 3}, 6], Null]
```



Operating the Hadamard gate on each qubit produces a superposition state consisting all logical basis states.

```
In[9]:= out = ReleaseHold[op] ** Ket[];
out // LogicalForm
Out[9]=  $\frac{|0_{S_1}0_{S_2}0_{S_3}\rangle}{2\sqrt{2}} + \frac{|0_{S_1}0_{S_2}1_{S_3}\rangle}{2\sqrt{2}} + \frac{|0_{S_1}1_{S_2}0_{S_3}\rangle}{2\sqrt{2}} + \frac{|0_{S_1}1_{S_2}1_{S_3}\rangle}{2\sqrt{2}} + \frac{|1_{S_1}0_{S_2}0_{S_3}\rangle}{2\sqrt{2}} + \frac{|1_{S_1}0_{S_2}1_{S_3}\rangle}{2\sqrt{2}} + \frac{|1_{S_1}1_{S_2}0_{S_3}\rangle}{2\sqrt{2}} + \frac{|1_{S_1}1_{S_2}1_{S_3}\rangle}{2\sqrt{2}}$ 
```

This show the same result in the quantum circuit model.

```
In[1]:= qc = QuantumCircuit[LogicalForm[Ket[], S@{1, 2, 3}], S[{1, 2, 3}, 6]]
ExpressionFor[qc] // LogicalForm
```

Out[1]=

$$\text{Out[1]} = \frac{|0_{S_1}0_{S_2}0_{S_3}\rangle}{2\sqrt{2}} + \frac{|0_{S_1}0_{S_2}1_{S_3}\rangle}{2\sqrt{2}} + \frac{|0_{S_1}1_{S_2}0_{S_3}\rangle}{2\sqrt{2}} + \frac{|0_{S_1}1_{S_2}1_{S_3}\rangle}{2\sqrt{2}} +$$

$$\frac{|1_{S_1}0_{S_2}0_{S_3}\rangle}{2\sqrt{2}} + \frac{|1_{S_1}0_{S_2}1_{S_3}\rangle}{2\sqrt{2}} + \frac{|1_{S_1}1_{S_2}0_{S_3}\rangle}{2\sqrt{2}} + \frac{|1_{S_1}1_{S_2}1_{S_3}\rangle}{2\sqrt{2}}$$

Let us compare it with an explicit construction. To do it first prepare the indices of the logical basis states in binary digits.

```
In[2]:= nn = Range[0, 2^3 - 1];
bit = IntegerDigits[nn, 2, 3]
Out[2]= {{0, 0, 0}, {0, 0, 1}, {0, 1, 0},
{0, 1, 1}, {1, 0, 0}, {1, 0, 1}, {1, 1, 0}, {1, 1, 1}}
```

This gives an explicit construction (unnormalized) of the superposition of all local basis states.

```
In[3]:= vec = Total[Ket[S@{1, 2, 3} \rightarrow #] & /@ bit];
vec // LogicalForm
Out[3]= |0_{S_1}0_{S_2}0_{S_3}\rangle + |0_{S_1}0_{S_2}1_{S_3}\rangle + |0_{S_1}1_{S_2}0_{S_3}\rangle +
|0_{S_1}1_{S_2}1_{S_3}\rangle + |1_{S_1}0_{S_2}0_{S_3}\rangle + |1_{S_1}0_{S_2}1_{S_3}\rangle + |1_{S_1}1_{S_2}0_{S_3}\rangle + |1_{S_1}1_{S_2}1_{S_3}\rangle
```

On other elements of the logical basis, the sign of each term is determined by the bitwise dot product of its bit-string with that of the input state.

```
In[4]:= in = Ket[S[{1, 2, 3}] \rightarrow {1, 0, 1}];
in = LogicalForm[in, S@{1, 2, 3}]
Out[4]= |1_{S_1}0_{S_2}1_{S_3}\rangle
```



```
In[5]:= out = ReleaseHold[op] ** in;
out // LogicalForm
Out[5]= \frac{|0_{S_1}0_{S_2}0_{S_3}\rangle}{2\sqrt{2}} - \frac{|0_{S_1}0_{S_2}1_{S_3}\rangle}{2\sqrt{2}} + \frac{|0_{S_1}1_{S_2}0_{S_3}\rangle}{2\sqrt{2}} - \frac{|0_{S_1}1_{S_2}1_{S_3}\rangle}{2\sqrt{2}} -
\frac{|1_{S_1}0_{S_2}0_{S_3}\rangle}{2\sqrt{2}} + \frac{|1_{S_1}0_{S_2}1_{S_3}\rangle}{2\sqrt{2}} - \frac{|1_{S_1}1_{S_2}0_{S_3}\rangle}{2\sqrt{2}} + \frac{|1_{S_1}1_{S_2}1_{S_3}\rangle}{2\sqrt{2}}
```

2.1.3 Rotations

Any unitary operator \hat{U} can always be written in the form $\hat{U} = \exp(-i\hat{H})$ with a Hermitian operator \hat{H} . On a two-dimensional vector space \mathcal{S} associated with a qubit, any Hermitian operator \hat{H} can be expanded in terms of the Pauli operators \hat{S}^μ as

$$\hat{H} = \phi_0 + \hat{S}^x B_x + \hat{S}^y B_y + \hat{S}^z B_z \quad (2.21)$$

where ϕ_0, B_x, B_y, B_z are real parameters. Regarding $\mathbf{B} := (B_x, B_y, B_z)$ as a three-dimensional vector, we consider the unit vector \mathbf{n} pointing to the same direction

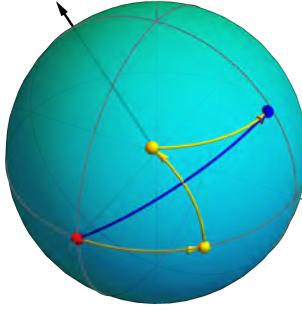


Figure 2.3: Visualization of transformations of states under the single-qubit operations. Up to a global phase factor, a single-qubit unitary operation is a rotation on the Bloch sphere. The rotation around the axis indicated by the black arrow is depicted by the blue arrow. The same rotation can be achieved by combining three rotations around y - or z -axis depicted by the yellow arrows.

as \mathbf{B} and another real parameter $\phi := 2|\mathbf{B}|$, where the factor 2 is just for later convenience. In terms of these new parameterization, \hat{H} reads as

$$\hat{H} = \phi_0 + \hat{\mathbf{S}} \cdot \mathbf{n} \phi/2, \quad (2.22)$$

where $\hat{\mathbf{S}} := (\hat{S}^x, \hat{S}^y, \hat{S}^z)$. In short, any unitary operator on \mathcal{S} has the form $\hat{U} = e^{-i\phi_0} \hat{U}_{\mathbf{n}}(\phi)$ with

$$\hat{U}_{\mathbf{n}}(\phi) := \exp(-i\hat{\mathbf{S}} \cdot \mathbf{n} \phi/2). \quad (2.23)$$

Here $e^{-i\phi_0}$ changes the global phase factor and is physically irrelevant. More important and interesting is the part $\hat{U}_{\mathbf{n}}(\phi)$, which as we will see below, describes a “rotation” around the axis \mathbf{n} by the angle ϕ . The rotations here are on the Bloch sphere—see Fig. 2.3 for an illustration—corresponding to the two-dimensional vector space \mathcal{S} , not in the real three-dimensional world. We will further denote the rotations around the μ -axis— \mathbf{n} parallel to the μ -axis—of the Bloch sphere by $\hat{U}_\mu(\phi)$.

To see that the unitary operator $\hat{U}_{\mathbf{n}}(\phi)$ in (2.23) corresponds to a rotation, recall that the Pauli operators \hat{S}^μ are the spin angular momentum operators of spin 1/2. That is, they are the generators of rotations and satisfy the commutation relations

$$[\hat{S}^\mu, \hat{S}^\nu] = 2i \sum_{\lambda} \hat{S}^\lambda \epsilon_{\lambda\mu\nu}. \quad (2.24)$$

The connection of the unitary operator $\hat{U}_\lambda(\phi)$ to rotation is seen more explicitly in the equivalent relation

$$\hat{U}_\lambda(\phi) \hat{S}^\nu \hat{U}_\lambda^\dagger(\phi) = \sum_{\mu} \hat{S}^\mu [R_\lambda(\phi)]_{\mu\nu}, \quad (2.25)$$

where $R_\lambda(\phi)$ is the 3×3 orthogonal matrix describing the rotation of three-dimensional coordinates around the λ -axis by angle ϕ .

The Pauli operators are generators of the rotational transformations in a two-dimensional complex vector space, and hence satisfy the fundamental commutation relations of angular momentum operators (up to a normalization factor).

```
In[1]:= op = S[All]
Out[1]= {Sx, Sy, Sz}

In[2]:= in = Outer[HoldForm@*Commutator, op, op];
out = ReleaseHold[in];
Thread[Flatten[in] → Flatten[out]] // TableForm
Out[2]/TableForm=
Commutator[Sx, Sx] → 0
Commutator[Sx, Sy] → 2 i Sz
Commutator[Sx, Sz] → -2 i Sy
Commutator[Sy, Sx] → -2 i Sz
Commutator[Sy, Sy] → 0
Commutator[Sy, Sz] → 2 i Sx
Commutator[Sz, Sx] → 2 i Sy
Commutator[Sz, Sy] → -2 i Sx
Commutator[Sz, Sz] → 0
```

In \mathbb{R}^3 , any 3×3 rotation matrix can be decomposed into three factors

$$R = R_z(\alpha)R_y(\beta)R_z(\gamma), \quad (2.26)$$

where α, β, γ are the so-called *Euler angles* and such a combination of rotations is called the *Euler rotation*. In the same manner, any unitary operator on \mathcal{S} can also be written as

$$\hat{U} = e^{-i\phi_0} \hat{U}_z(\alpha) \hat{U}_y(\beta) \hat{U}_z(\gamma), \quad (2.27)$$

that is, a combination of elementary “rotations” around the y - and z -axis and an additional overall phase shift. The unitary operator $\hat{U}(\alpha, \beta, \gamma) := \hat{U}_z(\alpha) \hat{U}_y(\beta) \hat{U}_z(\gamma)$ is called the Euler rotation in the two-dimensional vector space \mathcal{S} . Figure 2.3 illustrates an Euler rotation $\hat{U}(\pi/3, -\pi/3, \pi/4)$. It transforms $|v\rangle = (|0\rangle + |1\rangle)/\sqrt{2}$ (red dot in Fig. 2.3) is transformed to $|w\rangle = \hat{U}|v\rangle$ (blue dot in Fig. 2.3). The transformation is displayed by the blue arrow. The yellow dots and arrows depicts the transformations under $\hat{U}_z(\alpha)$, $\hat{U}_y(\beta)$, and $\hat{U}_z(\gamma)$, which combine to reproduce \hat{U} .

Consider a unitary operator represented by the following matrix.

```
In[3]:= mat = {
  {3 - I Sqrt[3], I - Sqrt[3]}, 
  {I + Sqrt[3], 3 + I Sqrt[3]}
} / 4;
mat // MatrixForm
Out[3]/MatrixForm=

$$\begin{pmatrix} \frac{1}{4} \times (3 - i\sqrt{3}) & \frac{1}{4} (i - \sqrt{3}) \\ \frac{1}{4} (i + \sqrt{3}) & \frac{1}{4} \times (3 + i\sqrt{3}) \end{pmatrix}$$

```

This gives its expression in terms of the Pauli operators.

```
In[5]:= op = Elaborate@ExpressionFor[mat, S]
Out[5]=  $\frac{3}{4} + \frac{i S^x}{4} - \frac{1}{4} i \sqrt{3} S^y - \frac{1}{4} i \sqrt{3} S^z$ 
```

```
In[6]:= Dagger[op] ** op
Out[6]= 1
```

This gives the Euler angles of the unitary operator.

```
In[7]:= angs = TheEulerAngles[mat]
Out[7]=  $\left\{\frac{\pi}{3}, \frac{\pi}{3}, 0\right\}$ 
```

Indeed, the Euler angles reproduces the original unitary operator.

```
In[8]:= new = EulerRotation[angs, S] // Elaborate
op - new
Out[8]=  $\frac{3}{4} + \frac{i S^x}{4} - \frac{1}{4} i \sqrt{3} S^y - \frac{1}{4} i \sqrt{3} S^z$ 
Out[9]= 0
```

Rotations $\hat{U}_z(\phi)$ around the z -axis in the Bloch space induces relative phase difference ϕ between the two logical basis states $|0\rangle$ and $|1\rangle$. In this sense, such rotations are called (relative) phase gates by phase angle ϕ . Two phase gates \hat{Q} and \hat{O} by angles $2\pi/4$ and $2\pi/8$, respectively, are particular common, and they are often called the *quadrant* and *octant phase gate*. In the logical basis, they are represented by the following diagonal matrices

$$\hat{Q} \doteq \begin{bmatrix} 1 & 0 \\ 0 & i \end{bmatrix}, \quad \hat{O} \doteq \begin{bmatrix} 1 & 0 \\ 0 & e^{i\pi/4} \end{bmatrix}, \quad (2.28)$$

respectively. Obviously, $\hat{O}^2 = \hat{Q}$ and $\hat{Q}^2 = \hat{Z}$.

Among (relative) phase gates, the quadrant and octant phase gates are most common.

```
In[10]:= qd = S[1, 7]
Matrix[qd] // MatrixForm
Out[10]=  $S_1^S$ 
Out[11]/MatrixForm=
 $\begin{pmatrix} 1 & 0 \\ 0 & i \end{pmatrix}$ 
```

```
In[12]:= qd ** qd
Out[12]=  $S_1^Z$ 
```

```
In[13]:= oc = S[1, 8]
Matrix[oc] // MatrixForm
Out[13]=  $S_1^T$ 
Out[14]/MatrixForm=
 $\begin{pmatrix} 1 & 0 \\ 0 & e^{\frac{i\pi}{4}} \end{pmatrix}$ 
```

```
In[15]:= oc ** oc
Out[15]=  $S_1^S$ 
```

2.2 Two-Qubit Gates

Let us next consider quantum logic gate operations acting on two qubits. Such operations are represented by 4×4 unitary matrices. We will see that any two-qubit gate operations can be decomposed into controlled- U gates. A controlled- U gate acts a unitary operator on one qubit depending on the logical state of the other qubit. A controlled- U gate on two qubits can be further decomposed into factors including only CNOT gate and single-qubit rotation gates. In this sense, the CNOT gate alone is sufficient for any two-qubit gate.

The controlled- U and CNOT gate have various interesting properties that make them useful in the implementation of quantum algorithms. In this section, we will first examine the basic properties of the CNOT gate, in particular, how it is used to generate an entanglement between two qubits. We then discuss the properties of the controlled- U gate and how to implement a controlled- U gate in terms of the CNOT gate and the single-qubit rotations. Finally, we discuss how an arbitrary two-qubit unitary operation can be decomposed into controlled- U gates.

2.2.1 CNOT, CZ, and SWAP

The CNOT or controlled-NOT gate is a quantum logic gate on two qubits that maps the logical basis states as

$$|c\rangle \otimes |t\rangle \mapsto |c\rangle \otimes |c \oplus t\rangle ; \quad c, t \in \{0, 1\} , \quad (2.29)$$

where the first qubit is typically called the *control qubit* (c) and the second qubit the *target qubit* (t). It has the following matrix representation in the logical basis

$$\text{CNOT} \doteq \begin{bmatrix} 1 & & & \\ & 1 & & \\ & & 0 & 1 \\ & & 1 & 0 \end{bmatrix} , \quad (2.30)$$

and expressed in terms of the Pauli operators on the control and target qubit as

$$\text{CNOT} = \frac{1}{2} \left(\hat{I} + \hat{S}_c^z + \hat{S}_t^z - \hat{S}_c^z \hat{S}_t^x \right) . \quad (2.31)$$

In a quantum circuit model, it is represented as the following circuit element:



where the smaller filled circle indicates the dependence on the state of the control qubit and the circled-plus sign denotes the conditional NOT action on the target qubit.

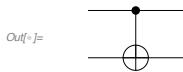
`CNOT[control, target]` denotes the CNOT gate in the quantum circuit model.

```
In[1]:= op = CNOT[S[1], S[2]]
```

```
Out[1]= CNOT[{S1}, {S2}]
```

This displays the quantum circuit model of the CNOT gate.

```
In[2]:= qc = QuantumCircuit[op]
```



This is the explicit expression of the CNOT gate in terms of the Pauli operators.

```
In[3]:= op = ExpressionFor[qc]
```

$$\frac{1}{2} - \frac{1}{2} S_1^z S_2^x + \frac{S_1^z}{2} + \frac{S_2^x}{2}$$

This is the matrix representation of the CNOT gate in the logical basis.

```
In[4]:= Matrix[op] // MatrixForm
```

```
Out[4]/MatrixForm=
```

$$\begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 1 & 0 \end{pmatrix}$$

This shows how the CNOT gate operates on the logical basis states.

```
In[5]:= in = Basis[S@{1, 2}];
```

```
out = op ** in;
```

```
Thread[in → out] // LogicalForm // TableForm
```

```
Out[5]/TableForm=
```

$$\begin{aligned} |0_{S_1} 0_{S_2}\rangle &\rightarrow |0_{S_1} 0_{S_2}\rangle \\ |0_{S_1} 1_{S_2}\rangle &\rightarrow |0_{S_1} 1_{S_2}\rangle \\ |1_{S_1} 0_{S_2}\rangle &\rightarrow |1_{S_1} 1_{S_2}\rangle \\ |1_{S_1} 1_{S_2}\rangle &\rightarrow |1_{S_1} 0_{S_2}\rangle \end{aligned}$$

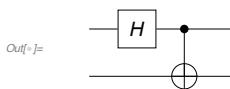
A simple yet important feature of CNOT is to copy the logical state of the control qubit to the target bit provided that the target bit is initially set to $|0\rangle$; or the reversed state when the target qubit is in $|1\rangle$. A vital implication is that CNOT generates an entangled state when the control qubit is in a superposition:

$$(|0\rangle c_0 + |1\rangle c_1) \otimes |0\rangle \mapsto |0\rangle \otimes |0\rangle c_0 + |1\rangle \otimes |1\rangle c_1. \quad (2.33)$$

Applying a single qubit rotation such the Hadamard gate on the control qubit prior to CNOT, one can thus generate entangled states from logical states. In this sense, such a circuit is called *quantum entangler circuit*.

As an example of the application of the CNOT gate, this shows an entangler quantum circuit.

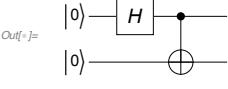
```
In[6]:= entangler = QuantumCircuit[S[1, 6], CNOT[S[1], S[2]]]
```



For example, when the input state is $|0\rangle \otimes |0\rangle$, the outcome of the circuit is given by one of the so-called Bell states.

This demonstrates the generation of an entangled state from a product state.

```
In[1]:= new = QuantumCircuit[LogicalForm[Ket[S@{1, 2} → {0, 0}], S@{1, 2}], entangler]
vec = ExpressionFor[new];
vec // LogicalForm
```



$$\text{Out}[1]= \frac{|0_{s_1} 0_{s_2}\rangle}{\sqrt{2}} + \frac{|1_{s_1} 1_{s_2}\rangle}{\sqrt{2}}$$

In general, the product states in the logical basis are transformed to the Bell states as you can see in the demonstration below.

This lists the mapping between the standard tensor-product basis states and the Bell states.

```
In[2]:= bs = Basis@S@{1, 2};
op = ExpressionFor[entangler];
out = op ** bs;
table = Thread[bs → out];
table // LogicalForm // TableForm
```

$$\text{Out}[2]//TableForm=$$

$ 0_{s_1} 0_{s_2}\rangle$	\rightarrow	$\frac{ 0_{s_1} 0_{s_2}\rangle}{\sqrt{2}} + \frac{ 1_{s_1} 1_{s_2}\rangle}{\sqrt{2}}$
$ 0_{s_1} 1_{s_2}\rangle$	\rightarrow	$\frac{ 0_{s_1} 1_{s_2}\rangle}{\sqrt{2}} + \frac{ 1_{s_1} 0_{s_2}\rangle}{\sqrt{2}}$
$ 1_{s_1} 0_{s_2}\rangle$	\rightarrow	$\frac{ 0_{s_1} 0_{s_2}\rangle}{\sqrt{2}} - \frac{ 1_{s_1} 1_{s_2}\rangle}{\sqrt{2}}$
$ 1_{s_1} 1_{s_2}\rangle$	\rightarrow	$\frac{ 0_{s_1} 1_{s_2}\rangle}{\sqrt{2}} - \frac{ 1_{s_1} 0_{s_2}\rangle}{\sqrt{2}}$

One can further generalize the above procedure to generate an maximally entangled states between larger systems: Consider two *quantum registers* each of which consisting of n qubits. We call them the “control” and “target” register, respectively, for the reason to get clear below. Upon applying the CNOT gate on each pair of the corresponding qubits in the two registers as the quantum circuit model in Fig. 2.4 (a), the logical basis states are transformed as

$$|c\rangle \otimes |t\rangle \mapsto |c\rangle \otimes |c \oplus t\rangle. \quad (2.34)$$

The above association rule is *formally* the same as the one in (2.29) for a single pair of qubits. Here, however, $|c\rangle := |c_1\rangle \otimes |c_2\rangle \otimes \dots \otimes |c_n\rangle$ and $|t\rangle := |t_1\rangle \otimes |t_2\rangle \otimes \dots \otimes |t_n\rangle$, and $c \oplus t$ denotes the bit-wise exclusive OR (or XOR),

$$c \oplus t := (c_1 \oplus t_1, c_2 \oplus t_2, \dots, c_n \oplus t_n), \quad (2.35)$$

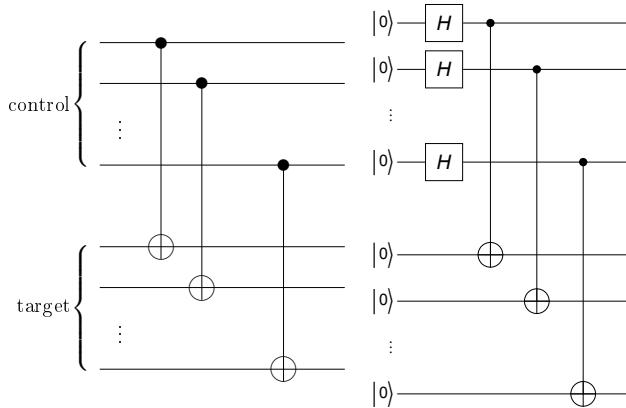


Figure 2.4: (a) A quantum circuit model which makes a copy of the logical state of the “control” quantum register to the “target” quantum register. The quantum circuit model transforms the logical basis states as $|c\rangle \otimes |t\rangle \mapsto |c\rangle \otimes |c \oplus t\rangle$, where c and t are b -bit strings. (b) A quantum circuit model generating a maximally entangled states between two quantum registers each of which consisting of n qubits.

for the n -bit strings $c \equiv (c_1, c_2, \dots, c_n)$ and $t \equiv (t_1, t_2, \dots, t_n)$. When the target register is prepared in the state $|0\rangle \equiv |0\rangle^{\otimes n}$, the logical basis state of the control register is copied to the target register,^{2.1}

$$|x\rangle \otimes |0\rangle \mapsto |x\rangle \otimes |x\rangle \quad (2.36)$$

under the set of CNOT gates on paired qubits. More interestingly, when the control register is prepared in a superposition, say, $2^{-n/2} \sum_{x=0}^{2^n-1} |x\rangle$, and the target register in the state $|0\rangle \equiv |0\rangle$, the transformation in (2.34) makes a copy of each logical state of the control register to the target register, and leads to a maximally entangled state,

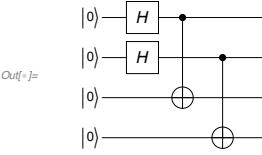
$$\frac{1}{2^{n/2}} \sum_{x=0}^{2^n-1} |x\rangle \otimes |0\rangle \mapsto |\Phi\rangle := \frac{1}{2^{n/2}} \sum_{x=0}^{2^n-1} |x\rangle \otimes |x\rangle \quad (2.37)$$

between the two registers (rather than single qubits). Since the superposition $2^{-n/2} \sum_{x=0}^{2^n-1} |x\rangle$ is obtained just by the Hadamard gates as shown in (2.18), the maximally entangled state $|\Phi\rangle$ in (2.37) can be generated from the logical state $|0\rangle \otimes |0\rangle$ through the quantum circuit model illustrated in Fig. 2.4 (b).

Here we want to generate a maximally entangled state between two quantum registers each of which consisting of two qubits.

^{2.1}It is important to note here that only *logical basis states* can be copied, but not a superposition of them. The latter is forbidden by the no-cloning theorem (Wooters & Zurek, 1982; Zurek, 2000).

```
In[5]:= qc = QuantumCircuit[LogicalForm[Ket[], S@{1, 2, 3, 4}],  
S[{1, 2}, 6], CNOT[S[1], S[3]], CNOT[S[2], S[4]],  
PlotRangePadding -> 0, ImagePadding -> {{36, 36}, {5, 5}}]
```



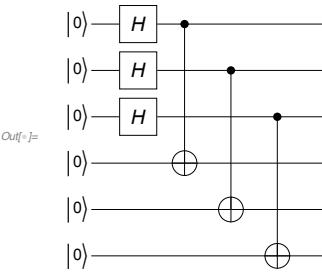
```
In[6]:= out = ExpressionFor[qc];  
out // LogicalForm  
Out[6]=  $\frac{1}{2} \left| 0_{S_1} 0_{S_2} 0_{S_3} 0_{S_4} \right\rangle + \frac{1}{2} \left| 0_{S_1} 1_{S_2} 0_{S_3} 1_{S_4} \right\rangle + \frac{1}{2} \left| 1_{S_1} 0_{S_2} 1_{S_3} 0_{S_4} \right\rangle + \frac{1}{2} \left| 1_{S_1} 1_{S_2} 1_{S_3} 1_{S_4} \right\rangle$ 
```

This example illustrate that the entanglement depends on the partition of the system. Indeed, the above system is a product state for the partition (1,3) and (2,4) qubits.

```
In[7]:= QuissoFactor[out]  
Out[7]=  $\frac{1}{2} (\left| 0_{S_1} 0_{S_3} \right\rangle + \left| 1_{S_1} 1_{S_3} \right\rangle) \otimes (\left| 0_{S_2} 0_{S_4} \right\rangle + \left| 1_{S_2} 1_{S_4} \right\rangle)$ 
```

The above construction can be generalized for a pair of n -qubit systems.

```
In[8]:= n = 3;  
qc = QuantumCircuit[LogicalForm[Ket[], S@Range[2 n]],  
S[Range[n], 6], Sequence @@ Table[CNOT[S[j], S[n+j]], {j, 1, n}]]
```



```
In[9]:= out = ExpressionFor[qc];  
out // LogicalForm  
Out[9]=  $\frac{\left| 0_{S_1} 0_{S_2} 0_{S_3} 0_{S_4} 0_{S_5} 0_{S_6} \right\rangle}{2\sqrt{2}} + \frac{\left| 0_{S_1} 0_{S_2} 1_{S_3} 0_{S_4} 0_{S_5} 0_{S_6} \right\rangle}{2\sqrt{2}} + \frac{\left| 0_{S_1} 1_{S_2} 0_{S_3} 0_{S_4} 1_{S_5} 0_{S_6} \right\rangle}{2\sqrt{2}} + \frac{\left| 0_{S_1} 1_{S_2} 1_{S_3} 0_{S_4} 1_{S_5} 1_{S_6} \right\rangle}{2\sqrt{2}} + \frac{\left| 1_{S_1} 0_{S_2} 0_{S_3} 1_{S_4} 0_{S_5} 0_{S_6} \right\rangle}{2\sqrt{2}} + \frac{\left| 1_{S_1} 0_{S_2} 1_{S_3} 1_{S_4} 0_{S_5} 1_{S_6} \right\rangle}{2\sqrt{2}} + \frac{\left| 1_{S_1} 1_{S_2} 0_{S_3} 1_{S_4} 1_{S_5} 0_{S_6} \right\rangle}{2\sqrt{2}} + \frac{\left| 1_{S_1} 1_{S_2} 1_{S_3} 1_{S_4} 1_{S_5} 1_{S_6} \right\rangle}{2\sqrt{2}}$ 
```

Another generalization of copying logical basis states to a series of qubits by consecutively applying CNOT gates with a single control qubit on multiple target qubits as in the quantum circuit model in Fig. 2.5 (a)—another equivalent quantum circuit model is given in Problem 2.5. When the target qubits are all prepared in the logical basis state $|0\rangle$, the series of CNOT gates makes the transformation

$$|x\rangle \otimes |0\rangle \otimes \cdots \otimes |0\rangle \mapsto |x\rangle \otimes |x\rangle \otimes \cdots \otimes |x\rangle , \quad (2.38)$$

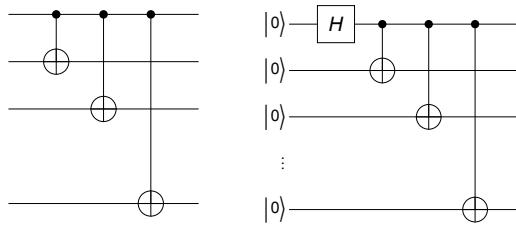


Figure 2.5: (a) Quantum circuit model copying the logical basis state of a single control qubit to multiple target qubits. (b) Quantum circuit model to generate the Greenberger-Horne-Zeilinger state among multiple qubits.

where $x = 0, 1$. Therefore, simply applying the Hadamard gate in the control qubit before the CNOT gates, one can generate the state of the form

$$|\text{GHZ}\rangle = \frac{|0\rangle \otimes |0\rangle \otimes \cdots \otimes |0\rangle + |1\rangle \otimes |1\rangle \otimes \cdots \otimes |1\rangle}{\sqrt{2}}. \quad (2.39)$$

The state in (2.39), which is known as the Greenberger-Horne-Zeilinger (GHZ) state ([Greenberger et al., 1989](#)), exhibits an entanglement of more than two particles. It has stimulated tests of non-locality of quantum mechanics beyond Bell inequalities ([Bouwmeester et al., 1999; Pan et al., 2000](#)).

Quantum entanglement is a valuable resource in quantum information processing and quantum communication. The most popular example is quantum teleportation to be discussed in Section 4.1. Inherently, the features elucidated in (2.36), (2.37) and (2.38)—and the related quantum circuit models in Figs. 2.4 and 2.5—will be frequently used in later parts of the book.

An interesting variant of CNOT gates is the so-called CZ or controlled-Z gate: It is a quantum logic gate on two qubits that maps the logical basis states as

$$\text{CZ} : |c\rangle \otimes |t\rangle \mapsto |c\rangle \otimes |t\rangle (-1)^{ct}. \quad (2.40)$$

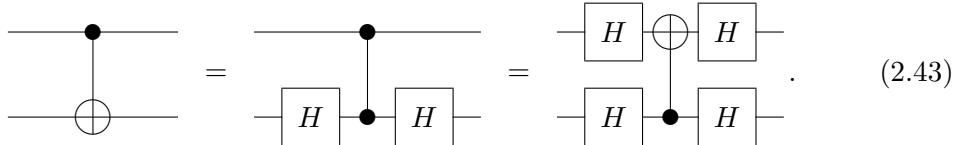
The matrix representation of the CZ gate in the logical basis is given by

$$\text{CZ} \doteq \begin{bmatrix} 1 & & & \\ & 1 & & \\ & & 1 & \\ & & & -1 \end{bmatrix}. \quad (2.41)$$

As it is symmetric for the two qubits, the distinction of the control and target qubit is meaningless. Accordingly, in the quantum circuit model, the gate is depicted by the following quantum circuit element



The filled circles on both qubit lines—rather than a square box on either qubit—indicates that the bit values of the both qubits remain unchanged. Noting the identity $\hat{H}\hat{Z}\hat{H} = \hat{X}$, one can regard that the CZ gate is equal to the CNOT gate up to the Hadamard gate on the target qubit. The relation between the CNOT and CZ gate is expressed in the following quantum circuit model



Depending on the Hamiltonian of particular physical system, the direct realization of the CNOT gate may be significantly difficult while the CZ gate is relatively easier to realize. In such a case, the identity (2.43) offers a straightforward workaround for the physical implementation of the CNOT gate.

The CZ (or controlled-Z) gate is a variant of the CNOT gate.

```
In[4]:= op = CZ[S[1], S[2]]
Out[4]= CZ[S1, S2]
```

This shows how it transforms the logical basis states.

```
In[5]:= bs = Basis@S@{1, 2};
out = op ** bs;
Thread[bs >> out] // LogicalForm // TableForm
Out[5]//TableForm=
|0S10S2> → |0S10S2>
|0S11S2> → |0S11S2>
|1S10S2> → |1S10S2>
|1S11S2> → -|1S11S2>
```

Here is the matrix representation of the CZ gate.

```
In[6]:= mat = Matrix[Elaborate@op];
mat // MatrixForm
Out[6]//MatrixForm=

$$\begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & -1 \end{pmatrix}$$

```

This is the quantum circuit model of the CZ gate.

```
In[7]:= cz = QuantumCircuit[CZ[S[1], S[2]]]
```



Note the following identity.

```
In[7]:= expr = HoldForm[S[1, 6] ** S[1, 3] ** S[1, 6] == S[1, 1]]
ReleaseHold[expr] // Elaborate
Out[7]= S1H ** S1Z ** S1H == S1X
Out[7]= True
```

It leads to the following relation between the CNOT gate and CZ gate.

```
In[8]:= new = QuantumCircuit[S[2, 6], cz, S[2, 6]]
Out[8]=
```

```
In[9]:= cnot = QuantumCircuit[CNOT[S[1], S[2]]]
Out[9]=
```

```
In[10]:= Elaborate[new - cnot]
Out[10]= 0
```

Another interesting two-qubit gate is the SWAP gate: The SWAP gate “swaps” the states of the two qubits, and maps the logical basis states as

$$\text{SWAP} : |x_1\rangle \otimes |x_2\rangle \mapsto |x_2\rangle \otimes |x_1\rangle . \quad (2.44)$$

As the states $|0\rangle \otimes |0\rangle$ and $|1\rangle \otimes |1\rangle$ are not altered by the operation, the matrix representation is given by

$$\text{SWAP} \doteq \begin{bmatrix} 1 & & & \\ & 0 & 1 & \\ & 1 & 0 & \\ & & & 1 \end{bmatrix} . \quad (2.45)$$

In the quantum circuit model, it is depicted as



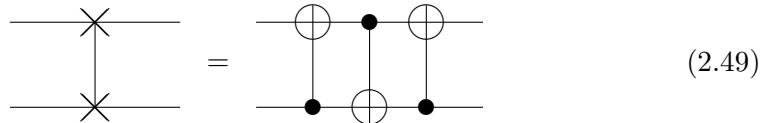
The SWAP gate can be implemented using the CNOT gate. To see this, first note that the simultaneous exchange of second and forth columns and rows of the matrix in (2.45) leads to

$$\begin{bmatrix} 1 & & & \\ & 1 & & \\ & & 0 & 1 \\ & & 1 & 0 \end{bmatrix} , \quad (2.47)$$

which is nothing but the matrix representation of the CNOT gate. On the other hand, the exchange of the second and forth columns and rows is described by the transformation matrix

$$\begin{bmatrix} 1 & & & \\ & 0 & & 1 \\ & & 1 & \\ 1 & & & 0 \end{bmatrix}, \quad (2.48)$$

which flips the bit values of the first qubit only when the second qubit is set to $|1\rangle$, and hence it corresponds to the CNOT gate with the second and first qubit as the control and target qubit, respectively. In short, the SWAP gate can be achieved by a combination of the CNOT gates as following



The SWAP gate exchanges the states of two qubits.

```
In[1]:= op = SWAP[S[1], S[2]]
Out[1]= SWAP[S1, S2]

In[2]:= bs = Basis@S@{1, 2};
new = op ** bs;
Thread[bs -> new] // LogicalForm // TableForm
Out[2]/TableForm=
|0S1 0S2> -> |0S1 0S2>
|0S1 1S2> -> |1S1 0S2>
|1S1 0S2> -> |0S1 1S2>
|1S1 1S2> -> |1S1 1S2>
```

This is the matrix representation of the SWAP gate in the logical basis.

```
In[3]:= Matrix[Elaborate@op] // MatrixForm
Out[3]/MatrixForm=
\begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix}
```

In the quantum circuit model, the SWAP gate is represented as following.

```
In[4]:= qc = QuantumCircuit[SWAP[S[1], S[2]]]
Out[4]=
```

The SWAP gate can be implemented by means of the CNOT gate.

```
In[5]:= new = QuantumCircuit[CNOT[S[1], S[2]], CNOT[S[2], S[1]], CNOT[S[1], S[2]]]
Out[5]=
```

```
In[7]:= Elaborate[qc - new]
Out[7]= 0
```

Interestingly, the SWAP gate itself is not universal, but the $\sqrt{\text{SWAP}}$ gate—the gate when squared equals to SWAP—is universal. That is, any quantum gate on a multi-qubit system can be implemented by combining $\sqrt{\text{SWAP}}$ and single-qubit rotations—see also Section 2.4 and Section 3.2.2. Indeed, one can combine the $\sqrt{\text{SWAP}}$ gate with single-qubit rotations to construct the CZ gate (Loss & DiVincenzo, 1998). As discussed above, the CZ gate just requires two more Hadamard gates to implement the CNOT gate and hence is universal.

Let us construct the CZ gate with the $\sqrt{\text{SWAP}}$ gate. This is the matrix representation of the the $\sqrt{\text{SWAP}}$ gate.

```
In[8]:= mat = MatrixPower[Matrix@Elaborate@SWAP[S[1], S[2]], 1/2];
mat // MatrixForm
Out[8]//MatrixForm=

$$\begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & \frac{1}{2} + \frac{i}{2} & \frac{1}{2} - \frac{i}{2} & 0 \\ 0 & \frac{1}{2} - \frac{i}{2} & \frac{1}{2} + \frac{i}{2} & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix}$$

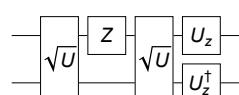
```

This is an explicit operator expression of the $\sqrt{\text{SWAP}}$ gate in terms of the Pauli operators.

```
In[9]:= sqrtSWAP = {ExpressionFor[mat, S[{1, 2}], , "Label" → "√U"];
Out[9]= { $\left(\frac{3}{4} + \frac{i}{4}\right) + \left(\frac{1}{4} - \frac{i}{4}\right) S_1^z S_2^z + \left(\frac{1}{2} - \frac{i}{2}\right) S_1^+ S_2^- + \left(\frac{1}{2} - \frac{i}{2}\right) S_1^- S_2^+$ , Null, Label →  $\sqrt{U}$ }
```

This is a quantum circuit model to construct the CZ gate from the $\sqrt{\text{SWAP}}$ gate and single-qubit gates. In this diagram, \sqrt{U} denotes the $\sqrt{\text{SWAP}}$ gate and U_z the rotation around the z-axis by angle $\pi/2$.

```
In[10]:= qc = QuantumCircuit[sqrtSWAP, S[1, 3], sqrtSWAP,
{Rotation[Pi/2, S[1, 3]], Rotation[-Pi/2, S[2, 3], "Label" → "U_z^\dagger"]}]
```



To check if it indeed implements the CZ gate, take a look at the matrix representation of the quantum circuit model.

```
In[11]:= Matrix[qc] // MatrixForm
Out[11]//MatrixForm=

$$\begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & -1 \end{pmatrix}$$

```

2.2.2 Controlled- U Gate

Consider two qubits, again called as the control and target qubits. Let \hat{U} be a unitary operator on the target qubit. The controlled- U gate is a unitary operator on the two-qubit Hilbert space defined analogously to CNOT by

$$\text{Ctrl}(\hat{U}) : |c\rangle \otimes |t\rangle \mapsto |c\rangle \otimes \hat{U}^c |t\rangle , \quad (2.50)$$

or equivalently, by

$$\text{Ctrl}(\hat{U}) := |0\rangle \langle 0| \otimes \hat{I} + |1\rangle \langle 1| \otimes \hat{U} . \quad (2.51)$$

If the control qubit is in $|0\rangle$, then it does nothing. On the other hand, if the control qubit is in $|1\rangle$, then it operates the unitary operator \hat{U} on the target qubit. In analogy with the CNOT gate, the matrix representation of the controlled- U gate is thus given by

$$\text{Ctrl}(\hat{U}) \doteq \begin{bmatrix} 1 & & & \\ & 1 & & \\ & & U_{11} & U_{12} \\ & & U_{21} & U_{22} \end{bmatrix} , \quad (2.52)$$

where U is the matrix representation of \hat{U} . In the quantum circuit model, a controlled- U gate is depicted as



The filled circle connected to the quantum circuit element on the target qubit indicates the conditional operation of the element conditioned on the state of the control qubit.

Consider a single-qubit rotation acting on the qubit `S[2, None]`. It is a rotation around the y-axis by angle ϕ .

```
In[7]:= U = Rotation[\phi, S[2, 2], "Label" \rightarrow "U"];
U // Elaborate // Matrix // MatrixForm
Out[7]= \left( \begin{array}{cc} \cos \left[ \frac{\phi}{2} \right] & -\sin \left[ \frac{\phi}{2} \right] \\ \sin \left[ \frac{\phi}{2} \right] & \cos \left[ \frac{\phi}{2} \right] \end{array} \right)
```

This shows the controlled- U gate.

```
In[8]:= qc = QuantumCircuit[ControlledU[S[1], U]]
```



This is the explicit expression of the controlled-U gate operation in terms of the Pauli operators.

```
In[7]:= op = Elaborate[qc]
Out[7]= Cos[\frac{\phi}{4}]^2 + S_1^z \ Sin[\frac{\phi}{4}]^2 + \frac{1}{2} \ i \ S_1^z \ S_2^y \ Sin[\frac{\phi}{2}] - \frac{1}{2} \ i \ S_2^y \ Sin[\frac{\phi}{2}]
```

The controlled-U gate maps the logical basis states as following.

```
In[8]:= bs = Basis[S@{1, 2}];
bs // LogicalForm
out = op ** bs;
out // LogicalForm
Out[8]= \{|\theta_{S_1}\theta_{S_2}\rangle, |\theta_{S_1}1_{S_2}\rangle, |1_{S_1}\theta_{S_2}\rangle, |1_{S_1}1_{S_2}\rangle\}
Out[9]= \{|\theta_{S_1}\theta_{S_2}\rangle, |\theta_{S_1}1_{S_2}\rangle, Cos[\frac{\phi}{2}] |1_{S_1}\theta_{S_2}\rangle + |1_{S_1}1_{S_2}\rangle \ Sin[\frac{\phi}{2}], 
Cos[\frac{\phi}{2}] |1_{S_1}1_{S_2}\rangle - |\theta_{S_1}\theta_{S_2}\rangle \ Sin[\frac{\phi}{2}]\}
```

To make the mapping clearer, this tabulates the above result.

```
In[10]:= new = QuissoFactor[#, S[1]] & /@ out;
Thread[bs \rightarrow new] // LogicalForm // TableForm
Out[10]//TableForm=
\begin{array}{l}
|\theta_{S_1}\theta_{S_2}\rangle \rightarrow |\theta_{S_1}\rangle \otimes |\theta_{S_2}\rangle \\
|\theta_{S_1}1_{S_2}\rangle \rightarrow |\theta_{S_1}\rangle \otimes |1_{S_2}\rangle \\
|1_{S_1}\theta_{S_2}\rangle \rightarrow |1_{S_1}\rangle \otimes \left(Cos[\frac{\phi}{2}] |\theta_{S_2}\rangle + |1_{S_2}\rangle \ Sin[\frac{\phi}{2}]\right) \\
|1_{S_1}1_{S_2}\rangle \rightarrow |1_{S_1}\rangle \otimes \left(Cos[\frac{\phi}{2}] |1_{S_2}\rangle - |\theta_{S_2}\rangle \ Sin[\frac{\phi}{2}]\right)
\end{array}
```

Let us take a look at the mapping more closely. When the first qubit is set to **Ket[0]**, it does nothing.

```
In[11]:= Let[Complex, c]
vec = Ket[] \times c[0] + Ket[S[2] \rightarrow 1] \times c[2];
LogicalForm[vec, S@{1, 2}]
Out[11]= c_0 |\theta_{S_1}\theta_{S_2}\rangle + c_2 |\theta_{S_1}1_{S_2}\rangle
In[12]:= new = op ** vec;
LogicalForm[new, S@{1, 2}]
Out[12]= c_0 |\theta_{S_1}\theta_{S_2}\rangle + c_2 |\theta_{S_1}1_{S_2}\rangle
```

When the control qubit -- the first qubit in this case -- is set to **Ket[1]**, it operates the unitary operator on the second qubit.

```
In[13]:= vec = Ket[S[1] \rightarrow 1] ** (Ket[] \times c[0] + Ket[S[2] \rightarrow 1] \times c[2]);
LogicalForm[vec, S@{1, 2}]
Out[13]= c_0 |1_{S_1}\theta_{S_2}\rangle + c_2 |1_{S_1}1_{S_2}\rangle
In[14]:= new = op ** vec;
LogicalForm[new, S@{1, 2}]
Out[14]= |1_{S_1}1_{S_2}\rangle \left(c_2 \ Cos[\frac{\phi}{2}] + c_0 \ Sin[\frac{\phi}{2}]\right) + |1_{S_1}\theta_{S_2}\rangle \left(c_0 \ Cos[\frac{\phi}{2}] - c_2 \ Sin[\frac{\phi}{2}]\right)
```

When the control qubit is in a superposition, the resulting state is an entangled state in general.

```
In[15]:= vec = Ket[] + Ket[S[1] \rightarrow 1];
LogicalForm[vec, S@{1, 2}]
Out[15]= |\theta_{S_1}\theta_{S_2}\rangle + |1_{S_1}\theta_{S_2}\rangle
```

```
In[5]:= new = op ** vec;
LogicalForm[new, S@{1, 2}]

Out[5]= |0_{S_1} 0_{S_2}\rangle + \text{Cos}\left[\frac{\phi}{2}\right] |1_{S_1} 0_{S_2}\rangle + |1_{S_1} 1_{S_2}\rangle \text{Sin}\left[\frac{\phi}{2}\right]
```

An important aspect of a controlled- U operator is that it induces relative phase shifts on the *control* qubit when the target is prepared in an eigenstate of \hat{U} . At a first glace, it may sound counter intuitive as the definition in (2.50) seems to indicate that it only changes the target qubit depending on the state of the control qubit and keeps the latter intact. This is another feature distinguishing quantum gates from the classical counterparts. Let us take a closer look to see how it works. Prepare the control qubit in a superposition $|\psi\rangle = |0\rangle + |1\rangle$ and the target qubit in a eigenstate $|u\rangle$ of \hat{U} with eigenvalue $e^{i\phi}$. The controlled- U gate transforms the state to

$$\begin{aligned} |\psi\rangle \otimes |u\rangle &\rightarrow |0\rangle \otimes |u\rangle + |1\rangle \otimes \hat{U}|u\rangle \\ &= |0\rangle \otimes |u\rangle + |1\rangle \otimes |u\rangle e^{i\phi} = (|0\rangle + |1\rangle e^{i\phi}) \otimes |u\rangle. \end{aligned} \quad (2.54)$$

This feature is extended to multi-qubit controlled- U gates and plays a crucial role in many quantum algorithms. In particular, the quantum phase estimation algorithm (Section 4.4) is a direct consequence of this feature.

When the target qubit is set to an eigenstate of the unitary operator, it does not change but the control qubit acquires the phase factor given by the eigenvalue of the target state.

```
In[6]:= vec = (Ket[] + Ket[S[1] -> 1]) ** (Ket[] - I Ket[S[2] -> 1]);
LogicalForm[QuissoFactor@vec, S@{1, 2}]

Out[6]= (|0_{S_1}\rangle + |1_{S_1}\rangle) \otimes (|0_{S_2}\rangle - \text{I} |1_{S_2}\rangle)

In[7]:= new = op ** vec // TrigToExp;
LogicalForm[QuissoFactor@new, S@{1, 2}]

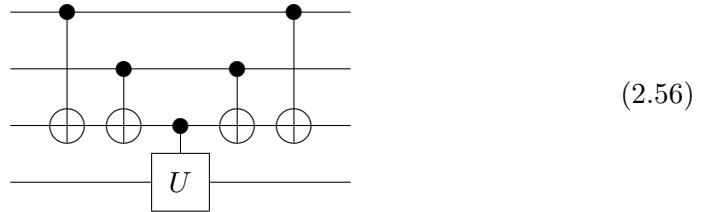
Out[7]= \left(|0_{S_1}\rangle + e^{\frac{i\phi}{2}} |1_{S_1}\rangle\right) \otimes (|0_{S_2}\rangle - \text{I} |1_{S_2}\rangle)
```

Combining the CNOT gate and the controlled- U gate, one can achieve a variety of conditional gate operations: For example, consider a system consisting of a control register of n qubits and a target register of a single qubit. Suppose that you want to operate a unitary gate \hat{U} on the target qubit only when an odd number of the control qubits is set to $|1\rangle$,

$$|c\rangle \otimes |t\rangle \mapsto |c\rangle \otimes \hat{U}^{c_1 \oplus \dots \oplus c_n} |t\rangle. \quad (2.55)$$

First operate the CNOT gates consecutively with the first $(n - 1)$ qubits in the control register as the control qubit and the last qubit in the control register as the target qubit. It transforms the n th qubit to $|c_1 \oplus \dots \oplus c_n\rangle$ —Problem 2.9. Then, by applying the controlled- U gate controlled by the n th qubit, the desired operation is implemented. To get the control qubits back to the original state,

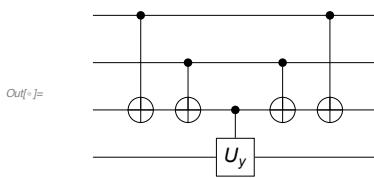
operate the CNOT gates in the reverse order. Overall, the following quantum circuit model implements the conditional operation



for the case of $n = 3$. This method is used for the implementation of multi-qubit controlled- U gate based on the Gray code—see Section 2.3.

This shows a quantum circuit model conditionally operating the logic gate U on the target qubit.

```
In[1]:= $n = 3;
cc = Table[CNOT[S[j], S[$n]], {j, 1, $n - 1}];
op = Rotation[\phi, S[$n + 1, 2]];
cU = ControlledU[S[$n], op];
qc = QuantumCircuit[Sequence @@ Flatten@{cc, cU, Reverse@cc}]
```



This shows how the above quantum circuit model maps the logical basis states. It affects the target qubit only when $c_1 \oplus c_2 \oplus c_3 = 1$.

```
In[=]:= ss = S[Range[$n], None];
bs = Basis[S@Range[$n + 1]];
out = Elaborate[qc] ** bs;
new = QuissoFactor[#, ss] & /@ out;
Thread[bs → new] // LogicalForm // TableForm
Out[=]/TableForm=
|0S10S20S30S4⟩ → |0S10S20S3⟩ ⊗ |0S4⟩
|0S10S20S31S4⟩ → |0S10S20S3⟩ ⊗ |1S4⟩
|0S10S21S30S4⟩ → |0S10S21S3⟩ ⊗ (Cos[φ/2] |0S4⟩ + |1S4⟩ Sin[φ/2])
|0S10S21S31S4⟩ → |0S10S21S3⟩ ⊗ (Cos[φ/2] |1S4⟩ - |0S4⟩ Sin[φ/2])
|0S11S20S30S4⟩ → |0S11S20S3⟩ ⊗ (Cos[φ/2] |0S4⟩ + |1S4⟩ Sin[φ/2])
|0S11S20S31S4⟩ → |0S11S20S3⟩ ⊗ (Cos[φ/2] |1S4⟩ - |0S4⟩ Sin[φ/2])
|0S11S21S30S4⟩ → |0S11S21S3⟩ ⊗ |0S4⟩
|0S11S21S31S4⟩ → |0S11S21S3⟩ ⊗ |1S4⟩
|1S10S20S30S4⟩ → |1S10S20S3⟩ ⊗ (Cos[φ/2] |0S4⟩ + |1S4⟩ Sin[φ/2])
|1S10S20S31S4⟩ → |1S10S20S3⟩ ⊗ (Cos[φ/2] |1S4⟩ - |0S4⟩ Sin[φ/2])
|1S10S21S30S4⟩ → |1S10S21S3⟩ ⊗ |0S4⟩
|1S10S21S31S4⟩ → |1S10S21S3⟩ ⊗ |1S4⟩
|1S11S20S30S4⟩ → |1S11S20S3⟩ ⊗ |0S4⟩
|1S11S20S31S4⟩ → |1S11S20S3⟩ ⊗ |1S4⟩
|1S11S21S30S4⟩ → |1S11S21S3⟩ ⊗ (Cos[φ/2] |0S4⟩ + |1S4⟩ Sin[φ/2])
|1S11S21S31S4⟩ → |1S11S21S3⟩ ⊗ (Cos[φ/2] |1S4⟩ - |0S4⟩ Sin[φ/2])
```

How can you implement a controlled- U gate? The operation involves only two qubits, and in principle, it should be possible to implement any specific controlled- U gate. However, as it will get clearer in Chapter 3, the requirements for physical implementation of two-qubit gates is far more difficult to fulfill on realistic systems than single-qubit gates. Fortunately, any controlled- U gate can be implemented using only CNOT gate and single-qubit gates. This is one of the basic steps in an establishment of the universal quantum computation.

Let \hat{U} a unitary gate on the second (target) qubit controlled by the first (control) qubit. Suppose that \hat{U} has Euler angles α , β , and γ and an additional phase factor $e^{i\varphi}$ [see (2.27)], $\hat{U} = e^{i\varphi}\hat{U}_z(\alpha)\hat{U}_y(\beta)\hat{U}_z(\gamma)$. Then one can find three unitary operators \hat{A} , \hat{B} , and \hat{C} such that

$$\hat{U} = e^{i\varphi}\hat{A}\hat{X}\hat{B}\hat{X}\hat{C}, \quad \hat{A}\hat{B}\hat{C} = \hat{I}, \quad (2.57)$$

where \hat{X} is the Pauli X operator. More explicitly, one common choice is

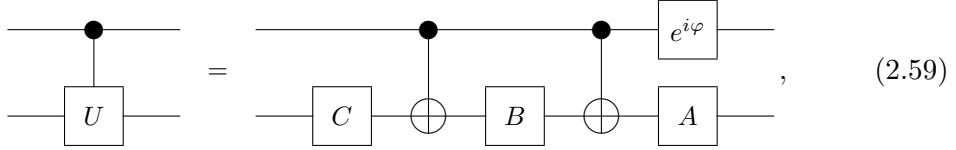
$$\hat{A} = \hat{U}_z(\alpha)\hat{U}_y(\beta/2), \quad (2.58a)$$

$$\hat{B} = \hat{U}_y(-\beta/2)\hat{U}_z(-(\alpha + \gamma)/2), \quad (2.58b)$$

$$\hat{C} = \hat{U}_z(-(\alpha - \gamma)/2). \quad (2.58c)$$

Since $\hat{U}_{y/z}(\phi) = \hat{X}\hat{U}_{y/z}(-\phi)\hat{X}$ for any ϕ , the above choice satisfies the desired properties in (2.57). The properties imply that the controlled- U gate can be

implemented as in the following quantum circuit model



where the last gate on the first qubit is the *relative* phase shift by φ , $|0\rangle\langle 0| + |1\rangle\langle 1|e^{i\varphi}$. Indeed, when the control qubit is in $|0\rangle$, the two CNOT gates in the middle do nothing, and the combined operator $\hat{A}\hat{B}\hat{C}$ on the target qubit ends up with the identity operator. With the control qubit in $|1\rangle$, on the other hand, the two CNOT gates are operational and the overall operator on the target qubit becomes $\hat{A}\hat{X}\hat{B}\hat{X}\hat{C} = e^{-i\varphi}\hat{U}$, where the phase factor is cancel by the opposite phase from the control qubit.

Consider a controlled-U gate.

```
In[1]:= matU = RandomUnitary[2];
matU // MatrixForm
Out[1]//MatrixForm=

$$\begin{pmatrix} -0.774044 - 0.632999 i & -0.00660915 + 0.0111572 i \\ -0.0126169 + 0.00299632 i & -0.374478 - 0.927145 i \end{pmatrix}$$

```

```
In[2]:= opU = ExpressionFor[matU, S[2]];
qc1 = QuantumCircuit[ControlledU[S[1], opU, "Label" \rightarrow "U"]]
```



For the decomposition, first find the Euler angles of the unitary operator.

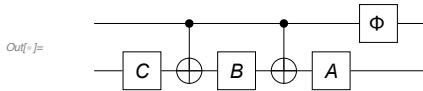
```
In[3]:= detU = Det[matU];
vphi = Arg[detU] / 2;
{a, b, c} = TheEulerAngles[matU / Sqrt[detU]]
Out[3]= {-0.918655, 0.0259364, -4.86309}
```

From the Euler angles, choose the component operators.

```
opA = EulerRotation[{a, b / 2, 0}, S[2], "Label" \rightarrow "A"];
opB = EulerRotation[{0, -b / 2, -(a + c) / 2}, S[2], "Label" \rightarrow "B"];
opC = EulerRotation[{0, 0, -(a - c) / 2}, S[2], "Label" \rightarrow "C"];
opD = Phase[vphi, S[1]];
```

Finally, construct the equivalent quantum circuit model.

```
In[4]:= qc2 = QuantumCircuit[opC, CNOT[S[1], S[2]], opB, CNOT[S[1], S[2]], opA, opD]
```



Check the result.

```
In[7]:= Elaborate[qc1 -> qc2] // Chop
Out[7]= 0
```

2.2.3 General Unitary Gate

Here we decompose an arbitrary two-qubit unitary gate into factors of controlled- U gates only. This is a remarkable advantage when one tries to build a quantum computer as you can just focus on how to realize the CNOT gate and single-qubit gates. Moreover, the same idea eventually leads to the proof of the universal quantum computation on a larger system, which will be discussed in Section 2.4.

Consider an arbitrary two-qubit unitary operator \hat{U} . In the logical basis, it is represented by a unitary matrix

$$U = \begin{bmatrix} U_{11} & U_{12} & U_{13} & U_{14} \\ U_{21} & U_{22} & U_{23} & U_{24} \\ U_{31} & U_{32} & U_{33} & U_{34} \\ U_{41} & U_{42} & U_{43} & U_{44} \end{bmatrix}. \quad (2.60)$$

To break the unitary operation \hat{U} down into more elementary quantum logic gates, we make use of *two-level unitary transformations*. A two-level unitary transformation is a unitary operation the matrix representation of which acts only on the two columns and rows of other matrices or column or row vectors. The descriptive word “two-level” should not be confused with “two-qubit”. A two-level transformation acts on multiple qubits, but it just transforms only two rows or columns at a time in the representation. For example, consider a two-level unitary transformation of the form

$$T_1 = \begin{bmatrix} 1 & & & \\ & 1 & & \\ & & \tilde{U}_{13}^* & \tilde{U}_{14} \\ & & \tilde{U}_{14}^* & -\tilde{U}_{13} \end{bmatrix}, \quad (2.61)$$

where $\tilde{U}_{ij} \propto U_{ij}$ with normalization factor (unspecified) such that $T_1^\dagger T_1 = T_1 T_1^\dagger = I$. When it multiplies U from the right, it does not change the first two columns of U . It only alters the last two columns; hence the name two-level transformation. The elements in the lower-right subblock of T_1 have been chosen so that the first element of the last column is canceled:

$$UT_1 = \begin{bmatrix} U_{11} & U_{12} & U'_{13} & 0 \\ U_{21} & U_{22} & U'_{23} & U'_{24} \\ U_{31} & U_{32} & U'_{33} & U'_{34} \\ U_{41} & U_{42} & U'_{43} & U'_{44} \end{bmatrix}. \quad (2.62)$$

Now take another two-level unitary transformation, this time, of the form

$$T_2 = \begin{bmatrix} 1 & & & \\ & \tilde{U}_{12}^* & \tilde{U}'_{13} & \\ & \tilde{U}'_{13}^* & -\tilde{U}_{12} & \\ & & & 1 \end{bmatrix}. \quad (2.63)$$

It removes U'_{13} while keeping the first and last column as they are:

$$UT_1T_2 = \begin{bmatrix} U_{11} & U''_{12} & 0 & 0 \\ U_{21} & U''_{22} & U''_{23} & U'_{24} \\ U_{31} & U''_{32} & U''_{33} & U'_{34} \\ U_{41} & U''_{42} & U''_{43} & U'_{44} \end{bmatrix}. \quad (2.64)$$

Similarly, we go further with the two-level unitary matrix

$$T_2 = \begin{bmatrix} \tilde{U}_{12}^* & \tilde{U}'_{13} & & \\ \tilde{U}'_{13}^* & -\tilde{U}_{12} & & \\ & & 1 & \\ & & & 1 \end{bmatrix}. \quad (2.65)$$

to remove the element U''_{12} and get

$$UT_1T_2T_3 = \begin{bmatrix} U'''_{11} & 0 & 0 & 0 \\ 0 & U'''_{22} & U''_{23} & U'_{24} \\ 0 & U'''_{32} & U''_{33} & U'_{34} \\ 0 & U'''_{42} & U''_{43} & U'_{44} \end{bmatrix}. \quad (2.66)$$

At this stage, all elements except for the first of the first column vanish — $U'''_{21} = U'''_{31} = U'''_{41} = 0$ —because the product $UT_1T_2T_3$ is a unitary matrix. Repeating the procedure, we can remove all off-diagonal elements to get

$$UT_1T_2 \dots T_L = I. \quad (2.67)$$

As T_j are all unitary, it enables us to rewrite U in the combination of two-level unitary transformation

$$U = T_L^\dagger \dots T_2^\dagger T_1^\dagger. \quad (2.68)$$

Consider a Hermitian operator on a two-qubit system. Physically, it corresponds to a transverse-field Ising model.

```
In[7]:= H = S[1, 1] ** S[2, 1] + S[1, 2] + S[2, 2] + S[1, 3] + S[2, 3]
matH = Matrix[H];
matH // MatrixForm
Out[7]//MatrixForm= S1^x S2^x + S1^y S2^z + S1^z S2^y + S2^z
```

$$\begin{pmatrix} 2 & -i & -i & 1 \\ i & 0 & 1 & -i \\ i & 1 & 0 & -i \\ 1 & i & i & -2 \end{pmatrix}$$

This is the unitary operator generated by the above Hermitian operator.

```
In[7]:= U = MultiplyExp[-I H Pi / 4]
Out[7]= e^(-\frac{1}{4} i \pi (S_1^x S_2^x + S_1^y S_2^y + S_1^z S_2^z))
```

This shows the matrix representation of U in the logical basis.

```
In[8]:= matU = Matrix[U] // Simplify;
matU // MatrixForm
```

$$\text{Out[8]//MatrixForm=}$$

$$\begin{pmatrix} \frac{1+5i}{2\sqrt{6}} & \frac{1-i}{\sqrt{2}} & \frac{1-i}{\sqrt{2}} & \frac{1-i}{\sqrt{2}} \\ \frac{1-i}{\sqrt{2}} & \frac{1+5i}{2\sqrt{6}} & -\frac{1+5i}{2\sqrt{6}} & \frac{1-i}{\sqrt{2}} \\ \frac{1-i}{\sqrt{2}} & -\frac{1+5i}{2\sqrt{6}} & \frac{1+5i}{2\sqrt{6}} & \frac{1-i}{\sqrt{2}} \\ \frac{1-i}{\sqrt{2}} & -\frac{1+5i}{2\sqrt{6}} & \frac{1-i}{\sqrt{2}} & -\frac{1+5i}{2\sqrt{6}} \\ \frac{1-i}{\sqrt{2}} & \frac{1-i}{\sqrt{2}} & \frac{1-i}{\sqrt{2}} & -\frac{1-i}{\sqrt{2}} \\ \frac{1-i}{\sqrt{2}} & \frac{1-i}{\sqrt{2}} & -\frac{1+5i}{2\sqrt{6}} & \frac{1-i}{\sqrt{2}} \\ \frac{1-i}{\sqrt{2}} & \frac{1-i}{\sqrt{2}} & \frac{1-i}{\sqrt{2}} & \frac{1-i}{\sqrt{2}} \end{pmatrix}$$

This shows the decomposition of the unitary matrix into two-level matrices (displaying the first three elements). For the purpose of efficiency, the resulting list is given in terms of `TwoLevelU`.

```
In[9]:= twl = TwoLevelDecomposition[matU] // Simplify;
twl[[;; 3]]
Out[9]= {TwoLevelU[{1, 0}, {0, 1}, {3, 4}, 4],
         TwoLevelU[{{\frac{1}{2} - \frac{3 i}{2}}, -{\frac{3}{2} + \frac{3 i}{2}}, {\frac{3}{2} - \frac{3 i}{2}}, {\frac{1}{2} + \frac{3 i}{2}}}, {3, 4}, 4],
         TwoLevelU[{{-\frac{1 - 3 i}{\sqrt{38}}}, \sqrt{\frac{14}{19}}, {-\sqrt{\frac{14}{19}}}, {-\frac{1 + 3 i}{\sqrt{38}}}}, {2, 3}, 4]}
```

For a more intuitive form, you can convert `TwoLevelU` into the normal matrix form using `Matrix`. Here the first three are shown.

```
In[10]:= MatrixForm /@ Matrix /@ twl[[;; 3]]
Out[10]= {{{1, 0, 0, 0}, {0, 1, 0, 0}, {0, 0, 1, 0}, {0, 0, 0, 1}}, {{1, 0, 0, 0}, {0, 1, 0, 0}, {0, 0, \frac{1+3i}{2\sqrt{2}}, 0}, {0, 0, \frac{3-3i}{2\sqrt{2}}, 0}}, {{1, 0, 0, 0}, {0, -\frac{1-3i}{\sqrt{38}}, \sqrt{\frac{14}{19}}, 0}, {0, -\sqrt{\frac{14}{19}}, -\frac{1+3i}{\sqrt{38}}, 0}, {0, 0, 0, 1}}}
```

Indeed, they reconstruct the original matrix.

```
In[11]:= new = Dot @@ Matrix /@ twl;
matU - new // Simplify // MatrixForm
```

$$\text{Out[11]//MatrixForm=}$$

$$\begin{pmatrix} 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \end{pmatrix}$$

We are still to express the two-level unitary transformation in terms of a controlled- U gate: The two-level unitary matrix—for example, T_1 in (2.61)—of the form

$$\begin{bmatrix} 1 & & & \\ & 1 & & \\ & & U_{11} & U_{12} \\ & & U_{21} & U_{22} \end{bmatrix}, \quad (2.69)$$

is already in the form of the matrix representation of a controlled- U gate in (2.52), and just corresponds to a single controlled- U :

$$\begin{bmatrix} 1 & & \\ & 1 & \\ & U_{11} & U_{12} \\ & U_{21} & U_{22} \end{bmatrix} = \begin{array}{c} \text{---} \bullet \text{---} \\ | \quad \quad \quad | \\ \text{---} \boxed{U} \text{---} \end{array}. \quad (2.70)$$

Consider a two-level matrix of the form.

```
In[~]:= matU = TheHadamard[];
code = TwoLevelU[matU, {3, 4}, 4];
full = Matrix[code];
full // MatrixForm
```

Out[~]:= $\begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & \frac{1}{\sqrt{2}} & \frac{1}{\sqrt{2}} \\ 0 & 0 & \frac{1}{\sqrt{2}} & -\frac{1}{\sqrt{2}} \end{pmatrix}$

It corresponds to a single controlled- U gate.

```
In[~]:= ctrlU = GrayTwoLevelU[matU, {3, 4}, S@{1, 2}];
QuantumCircuit[ctrlU]
```

Out[~]:= $\begin{array}{c} \text{---} \bullet \text{---} \\ | \quad \quad \quad | \\ \text{---} \boxed{U} \text{---} \end{array}$

A two-level unitary matrix of the form

$$\begin{bmatrix} 1 & & & \\ & U_{11} & U_{12} & \\ & & 1 & \\ & U_{21} & U_{22} & \end{bmatrix} \quad (2.71)$$

also corresponds to a single controlled- U gate with the control and target qubit exchanged—here the first qubit is the target qubit and the second the control qubit:

$$\begin{bmatrix} 1 & & & \\ & U_{11} & U_{12} & \\ & & 1 & \\ & U_{21} & U_{22} & \end{bmatrix} = \begin{array}{c} \text{---} \boxed{U} \text{---} \\ | \bullet | \\ \text{---} \end{array}. \quad (2.72)$$

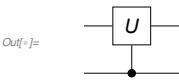
Consider a two-level matrix of the form.

```
In[=]:= matU = TheHadamard[];
code = TwoLevelU[matU, {2, 4}, 4];
full = Matrix[code];
full // MatrixForm
```

$$\text{Out}[=]/\text{MatrixForm}= \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & \frac{1}{\sqrt{2}} & 0 & \frac{1}{\sqrt{2}} \\ 0 & 0 & 1 & 0 \\ 0 & \frac{1}{\sqrt{2}} & 0 & -\frac{1}{\sqrt{2}} \end{pmatrix}$$

It corresponds to a single controlled-U gate with the control and target qubit exchanged.

```
In[=]:= ctrlU = GrayTwoLevelU[matU, {2, 4}, S@{1, 2}];
QuantumCircuit[ctrlU]
```



More complicated is the two-level unitary matrix of the form

$$\begin{bmatrix} 1 & & & \\ & U_{11} & U_{12} & \\ & U_{21} & U_{22} & \\ & & & 1 \end{bmatrix}. \quad (2.73)$$

As it affects both qubits simultaneously, it cannot be represented by a single controlled- U gate. However, it is possible to bring it to the form (2.70) by exchanging the second and forth columns and rows. The specified exchanges correspond to flipping the bit values of the first qubit only when the second qubit has the value 1, that is, the CNOT gate with the second qubit as the control qubit and the first qubit as the target qubit. Through the exchange, the unitary matrix U itself is modified and the rows and columns are exchanged, which corresponds to the basis change by the Pauli X matrix, $U \rightarrow U' = XUX$. Putting all together, the two-level unitary matrix is implemented as

$$\begin{bmatrix} 1 & & & \\ & U_{11} & U_{12} & \\ & U_{21} & U_{22} & \\ & & & 1 \end{bmatrix} = \begin{array}{c} \text{---} \oplus \text{---} \cdot \text{---} \oplus \text{---} \\ | \qquad | \qquad | \qquad | \\ \bullet \qquad \text{---} \qquad \text{---} \qquad \bullet \\ | \qquad | \qquad | \qquad | \\ \text{---} \qquad U' \qquad \text{---} \end{array}. \quad (2.74)$$

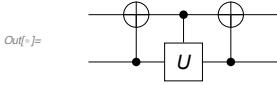
Consider a two-level matrix of the form.

```
In[=]:= matU = TheHadamard[];
code = TwoLevelU[matU, {2, 3}, 4];
full = Matrix[code];
full // MatrixForm
```

$$\text{Out}[=]/\text{MatrixForm}= \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & \frac{1}{\sqrt{2}} & \frac{1}{\sqrt{2}} & 0 \\ 0 & \frac{1}{\sqrt{2}} & -\frac{1}{\sqrt{2}} & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix}$$

In this case, we need to apply the CNOT gate before and after the controlled-U gate.

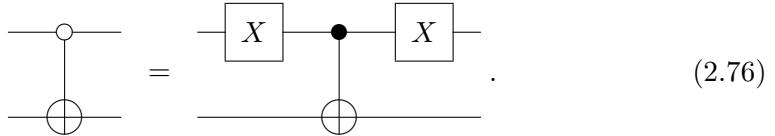
```
In[7]:= ctrlU = GrayTwoLevelU[matU, {2, 3}, S@{1, 2}];  
QuantumCircuit[ctrlU]
```



In a similar manner, we can implement another form of two-level unitary matrix as

$$\begin{bmatrix} U_{11} & & U_{12} \\ & 1 & \\ U_{21} & & U_{22} \end{bmatrix} = \text{QuantumCircuit}[U] . \quad (2.75)$$

Here we have adopted a short-hand diagram for the modified-CNOT gate, which flips the bit value of the target qubit when the control qubit is in the state $|0\rangle$ rather than $|1\rangle$. It is achieved by operating the Pauli X gate before and after the usual CNOT gate,

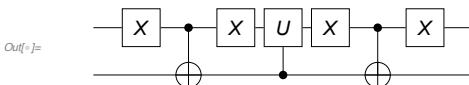


Consider a two-level matrix of the form.

```
In[8]:= matU = TheHadamard[];  
code = TwoLevelU[matU, {1, 4}, 4];  
full = Matrix[code];  
full // MatrixForm  
Out[8]//MatrixForm=
```

$$\begin{pmatrix} \frac{1}{\sqrt{2}} & 0 & 0 & \frac{1}{\sqrt{2}} \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ \frac{1}{\sqrt{2}} & 0 & 0 & -\frac{1}{\sqrt{2}} \end{pmatrix}$$

```
In[9]:= ctrlU = GrayTwoLevelU[matU, {1, 4}, S@{1, 2}];  
QuantumCircuit@@ctrlU
```



So far, we have figured out the implementations of 4×4 two-level unitary matrices of different forms just by simple inspection. For more than two qubits, the size of the two-level unitary matrices is much larger and it is difficult to find proper implementations in such a way. Fortunately, there is a systematic way based on the Gray code, which will be discussed later in Section 2.4.

In summary, an arbitrary two-qubit unitary gate can be carried out by first decomposing its matrix representation into two-level unitary matrices and then implementing the two-level unitary matrices by means of the CNOT gate and the controlled- U gate.

Let us consider again the two-qubit model demonstrated before.

```
In[7]:= H = S[1, 1] ** S[2, 1] + S[1, 2] + S[2, 2] + S[1, 3] + S[2, 3]
U = MultiplyExp[-I H Pi / 4]
matU = Matrix[U];
Out[7]= S_1^x S_2^x + S_1^y + S_1^z + S_2^y + S_2^z
Out[7]= E^-I \pi (S_1^x S_2^x + S_1^y + S_1^z + S_2^y + S_2^z)
```

This is the decomposition into the two-level matrices, just showing the first four.

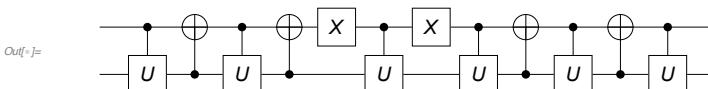
```
In[8]:= twl = TwoLevelDecomposition[matU] // Simplify;
MatrixForm /@ Matrix /@ twl[[;; 3]]
Out[8]= {{{1, 0, 0, 0}, {0, 1, 0, 0}, {0, 0, 1, 0}, {0, 0, 0, 1}}, {{1, 0, 0, 0}, {0, 1, 0, 0}, {0, 0, 2/7 (-1/2 + 3 I), -2/7 (-1/2 + 3 I)}, {0, 0, 2/7 (3/2 + 3 I), 2/7 (3/2 + 3 I)}}, {{1, 0, 0, 0}, {0, -1/7 Sqrt[19], Sqrt[14/19], 0}, {0, -Sqrt[14/19], -1/7 Sqrt[38], 0}, {0, 0, 0, 1}}}
```

The two-level matrices are written in terms of the controlled- U and CNOT gate, again showing the first four. The first of the list twl happens to be the identity matrix in this case, and it is excluded in the further analysis.

```
In[9]:= gates = GrayTwoLevelU[#, S@{1, 2}] & /@ Rest[twl];
gates[[;; 3]]
Out[9]= {{ControlledU[{S1}, 1/(2 Sqrt[7]) - 3 I S2^z/(2 Sqrt[7]) - ((3/2 + 3 I/2) S2^+ + (3/2 - 3 I/2) S2^-)/Sqrt[7], Label -> U]}, {CNOT[{S2}, {S1}], ControlledU[{S1}, -1/Sqrt[38] - I Sqrt[14/19]/19 S2^y - 3 I S2^z/Sqrt[38], Label -> U], CNOT[{S2}, {S1}]}, {ControlledU[{S1}, -5/(14 Sqrt[2]) - 5 I S2^z/(14 Sqrt[2]) + 3/Sqrt[19] S2^+ - 3/(14 Sqrt[2]) S2^-, Label -> U]}}
```

This shows the overall quantum circuit model. Here the same label “ U ” has been used for different controlled- U gates just for convenience. Do not forget to reverse the order before plugging the gates in the quantum circuit model.

```
In[10]:= qc = QuantumCircuit @@ Reverse @ Flatten[gates]
```



Check the above quantum circuit by converting it into the matrix representation.

```
In[=]:= new = Matrix[qc] // Simplify;
new // MatrixForm
Out[=]//MatrixForm=

$$\begin{pmatrix} -\frac{1}{2} + \frac{5i}{6} & \frac{1}{6} - \frac{i}{2} & \frac{1}{6} - \frac{i}{2} & \frac{1}{2} - \frac{i}{2} \\ \frac{1}{\sqrt{2}} & \frac{1}{\sqrt{2}} & \frac{1}{\sqrt{2}} & \frac{1}{\sqrt{2}} \\ \frac{1}{6} - \frac{i}{2} & \frac{1}{2} + \frac{i}{6} & -\frac{1}{2} + \frac{5i}{6} & \frac{1}{2} + \frac{i}{2} \\ \frac{\sqrt{2}}{2} & \frac{\sqrt{2}}{2} & -\frac{\sqrt{2}}{2} & \frac{\sqrt{2}}{2} \\ \frac{1}{6} - \frac{i}{2} & -\frac{1}{2} + \frac{5i}{6} & \frac{1}{2} + \frac{i}{6} & -\frac{1}{2} + \frac{i}{2} \\ \frac{\sqrt{2}}{2} & -\frac{\sqrt{2}}{2} & \frac{\sqrt{2}}{2} & -\frac{\sqrt{2}}{2} \\ \frac{1}{2} - \frac{i}{2} & \frac{1}{2} - \frac{i}{2} & \frac{1}{2} - \frac{i}{2} & -\frac{1}{2} - \frac{i}{2} \\ \frac{\sqrt{2}}{2} & \frac{\sqrt{2}}{2} & \frac{\sqrt{2}}{2} & -\frac{\sqrt{2}}{2} \end{pmatrix}$$

In[=]:= new - matU // Simplify // MatrixForm
Out[=]//MatrixForm=

$$\begin{pmatrix} 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \end{pmatrix}$$

```

2.3 Multi-Qubit Controlled Gates

Let $\mathcal{S}^{\otimes m}$ and $\mathcal{S}^{\otimes n}$ be the Hilbert spaces of the control and target register consisting of m and n qubits, respectively. Suppose that \hat{U} is a unitary operator on the target register. The multi-qubit controlled- U operator is defined by

$$\text{Ctrl}(\hat{U}) : |c\rangle \otimes |t\rangle \mapsto |c\rangle \otimes \hat{U}^{c_1 c_2 \dots c_m} |t\rangle , \quad (2.77)$$

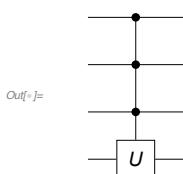
where $c \equiv (c_1 c_2 \dots c_m)_2$. The unitary transformation \hat{U} acts on the target qubits only when every control qubit is set to $|1\rangle$. We will be most interested in the case with $n = 1$, where the matrix representation takes the form

$$\text{Ctrl}(\hat{U}) \doteq \begin{bmatrix} 1 & & & \\ & 1 & & \\ & & \ddots & \\ & & & U_{11} & U_{12} \\ & & & U_{21} & U_{22} \end{bmatrix}. \quad (2.78)$$

It is the prototype form of a two-level unitary matrix on $(m + 1)$ qubits. Indeed, any two-level unitary matrix can be put into this form by exchanging columns and rows—equivalent to basis changes. Multi-qubit controlled- U gates thus arise naturally as we discuss the universal quantum computation in Section 2.4.

This is a three-qubit controlled-U gate.

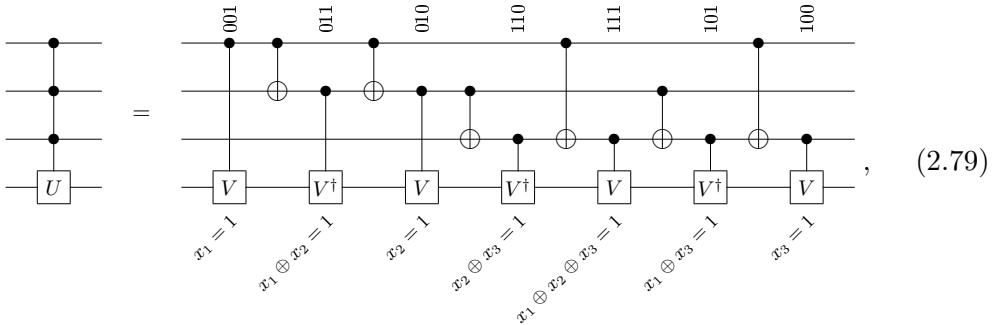
```
In[=]:= qc = QuantumCircuit[ControlledU[S@{1, 2, 3}, S[4, 1], "Label" -> "U"]]
```



A reliable implementation of multi-qubit controlled- \hat{U} gate is essential in many quantum algorithms. A notable example is the quantum oracle (see Section 4.2.1), which is a key component of quantum decision problems as well as quantum search problems. Here we introduce some widely known methods to implement it efficiently.

2.3.1 Gray Code

We first discuss a systematic method based on the Gray code to decompose a multi-qubit controlled- U gate into factors of either the single-qubit controlled- U or CNOT gate: For example, consider a 3-qubit controlled- U gate as shown in the following quantum circuit model (Barenco *et al.*, 1995)



where \hat{V} is another unitary operator such that $\hat{V}^4 = \hat{U}$. When every control qubit is set to $|1\rangle$, all \hat{V} gates in the diagram take effects on the target qubit while \hat{V}^\dagger gates are ineffective. To examine the case with the control qubits in a general state $|x_1\rangle \otimes \dots \otimes |x_n\rangle$, note that the gates on the target qubit are effective under the conditions specified in terms of the bit values at the bottom of the diagram—see also Eq. (2.56). The conditions are fulfilled systematically by following the Gray code sequence,^{2.2} the strings of which are indicated at the top of the diagram. The identity for the bitwise AND,

$$\begin{aligned} 2^{n-1}(x_1 x_2 \cdots x_n) &= \sum_{k_1} x_{k_1} - \sum_{k_1 < k_2} (x_{k_1} \oplus x_{k_2}) \\ &+ \sum_{k_1 < k_2 < k_3} (x_{k_1} \oplus x_{k_2} \oplus x_{k_3}) - \cdots + (-1)^{n-1}(x_1 \oplus x_2 \oplus \cdots \oplus x_n), \end{aligned} \quad (2.80)$$

ensures that the quantum circuit model on the right-hand side of (2.79) reproduces the desired multi-qubit controlled- U gate on the left-hand side.

In general, a n -qubit controlled- U gate can be implemented by combining 2^{n-1} controlled- V gates, $(2^{n-1} - 1)$ controlled- V^\dagger gates, and $(2^n - 2)$ CNOT

^{2.2}The *Gray code sequence* is an arrangement of bits such that two successive values differ in only one bit.

gates, where \hat{V} is a unitary operator satisfying $\hat{V}^{2^n-1} = \hat{U}$. For relatively small n ($n \leq 8$), the Gray code is known to be the most efficient method. However, it grows exponentially and eventually becomes impractical. For those cases, several methods have been proposed where the computational cost increases quadratically with the size of the quantum register (Barenco *et al.*, 1995; Nielsen & Chuang, 2011).

Consider a three-qubit register as an example.

```
$L = 3;
jj = Range[$L];
cc = S[jj, None];
```

This is the gate to operate on the target qubit.

```
In[7]:= T = S[$L + 1, None];
op = Rotation[Pi/3, T[1]] // Elaborate
Out[7]= 
$$\frac{\sqrt{3}}{2} - \frac{i}{2} S_4^x$$

```

This decomposes the multi-qubit controlled-U gate based on the Gray code -- showing only a part of it. Each component is either CNOT or a two-qubit controlled-U gate.

```
In[8]:= gc = GrayControlledU[cc, op]; gc // ;; 3]
Out[8]= 
$$\left\{ \begin{aligned} &\text{ControlledU}\left[\{S_1\}, \right. \\ &\left. \frac{2^{1/4} e^{-\frac{i\pi}{24}} + 2^{1/4} e^{\frac{i\pi}{24}}}{2 \cdot 2^{1/4}} + \frac{\left(2^{1/4} e^{-\frac{i\pi}{24}} - 2^{1/4} e^{\frac{i\pi}{24}}\right) S_4^+}{2 \times 2^{1/4}} + \frac{\left(2^{1/4} e^{-\frac{i\pi}{24}} - 2^{1/4} e^{\frac{i\pi}{24}}\right) S_4^-}{2 \cdot 2^{1/4}}, \text{Label} \rightarrow V \right], \\ &\text{CNOT}\left[\{S_1\}, \{S_2\}\right], \text{ControlledU}\left[\{S_2\}, \right. \\ &\left. \frac{2^{1/4} e^{-\frac{i\pi}{24}} + 2^{1/4} e^{\frac{i\pi}{24}}}{2 \cdot 2^{1/4}} + \frac{\left(-2^{1/4} e^{-\frac{i\pi}{24}} + 2^{1/4} e^{\frac{i\pi}{24}}\right) S_4^+}{2 \times 2^{1/4}} + \frac{\left(-2^{1/4} e^{-\frac{i\pi}{24}} + 2^{1/4} e^{\frac{i\pi}{24}}\right) S_4^-}{2 \times 2^{1/4}}, \text{Label} \rightarrow V^\dagger \right] \end{aligned} \right\}$$

```

This is a quantum circuit model of the decomposition.

```
In[9]:= qc = QuantumCircuit[gc]
```

Finally check the result.

```
In[10]:= expr = ExpressionFor[qc];
expr2 = ControlledU[cc, op] // Elaborate;
expr - expr2 // Simplify
Out[10]= 0
```

2.3.2 Multi-Qubit Controlled-NOT

The multi-qubit controlled-NOT gate is an important special case of the multi-qubit controlled- U gate, where the unitary operator \hat{U} equals to the Pauli \hat{X} . It transforms the logical basis states as [see Eq. (2.77)]

$$|c\rangle \otimes |t\rangle \mapsto |c\rangle \otimes \hat{X}^{c_1 c_2 \cdots c_m} |t\rangle , \quad (2.81a)$$

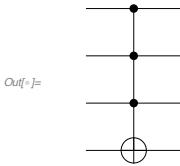
or equivalently,

$$|c\rangle \otimes |t\rangle \mapsto |c\rangle \otimes |(c_1 c_2 \cdots c_n) \oplus t\rangle . \quad (2.81b)$$

The multi-qubit controlled-NOT gate commonly occurs when one converts the two-level unitary transformation—see Section 2.2.3—on more than two qubits. Indeed, we will generalize the procedure and discuss a systematic way of conversion based on the Gray code in Section 2.4.

This shows a generalized CNOT gate, controlled by three qubits rather than a single qubit.

```
In[7]:= qc = QuantumCircuit[CNOT[S@{1, 2, 3}, S[4]]]
```



Here is the explicit expression of the multi-qubit CNOT gate in terms of the Pauli operators.

```
In[8]:= op = ExpressionFor[qc]
Out[8]= 
$$\frac{7}{8} - \frac{1}{8} S_1^z S_2^z - \frac{1}{8} S_1^z S_3^z - \frac{1}{8} S_1^z S_4^z - \frac{1}{8} S_2^z S_3^z - \frac{1}{8} S_2^z S_4^z - \frac{1}{8} S_3^z S_4^z + \frac{1}{8} S_1^z S_2^z S_3^z + \frac{1}{8} S_1^z S_2^z S_4^z + \frac{1}{8} S_1^z S_3^z S_4^z + \frac{1}{8} S_2^z S_3^z S_4^z - \frac{1}{8} S_1^z S_2^z S_3^z S_4^z + \frac{S_1^z}{8} + \frac{S_2^z}{8} + \frac{S_3^z}{8} + \frac{S_4^z}{8}$$

```

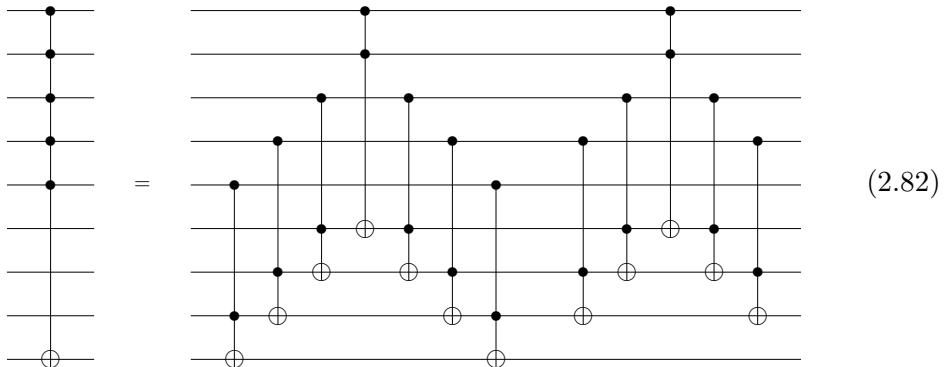
Here we compare it with the expression explicitly constructed in terms of a projection operator.

```
In[9]:= prj = Multiply @@ S[{1, 2, 3}, 11];
new = (1 - prj) + prj ** S[4, 1]
op - new // Elaborate
Out[9]= 
$$1 - (|1\rangle\langle 1|)_{S_1} (|1\rangle\langle 1|)_{S_2} (|1\rangle\langle 1|)_{S_3} + (|1\rangle\langle 1|)_{S_1} (|1\rangle\langle 1|)_{S_2} (|1\rangle\langle 1|)_{S_3} S_4^z$$

Out[9]= 0
```

An efficient implementation of a multi-qubit CNOT gate uses additional qubits not directly involved in the gate operation itself as in the following quantum

circuit model (Barenco *et al.*, 1995):

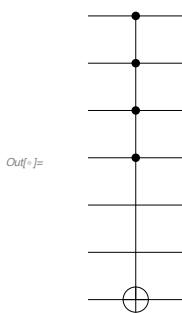


It is emphasized that the quantum states of the extra qubits should not be confused with the so-called “ancillary qubits” in the sense that they do not have to be initialized in a certain fixed quantum state and their state is not altered. The desired gate operation is performed properly regardless of the initial state of the extra qubits and their quantum state is restored after the gate operation.

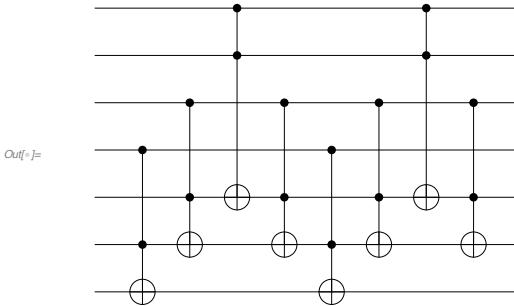
Here we demonstrate a multi-qubit CNOT gate.

```
$n = 4;
$m = 2;
cc = Range[$n];
aa = Range[$n + 1, $n + $m];
tt = $n + $m + 1;
```

In[7]:= qc = QuantumCircuit[CNOT[S[cc], S[tt]], "Visible" -> S[aa]]



```
In[5]:= tofa = Table[Toffoli[S[$n - j + 1], S[$n + $m - j + 1], S[tt - j + 1]], {j, 1, $m}];  
tofb = Toffoli[S[1], S[2], S[$n + 1]];  
tofc = Rest@tofa;  
new = QuantumCircuit[  
  Sequence @@ Flatten@{tofa, tofb, Reverse@tofa, tofc, tofb, Reverse@tofc}]
```



```
In[6]:= Timing[Elaborate[qc - new]]  
Out[6]= {12.0363, 0}
```

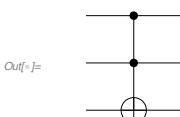
In the quantum circuit model (2.82), we have introduced another special case of multi-qubit CNOT gates, which is called the *Toffoli gate*, denoted by the quantum circuit element



The Toffoli gate has attracted interested as it is univesal for classical reversible computation. Unfortunately, however, it is not universal for quantum computation.

This is a quantum circuit model of the Toffoli gate.

```
In[7]:= qc = QuantumCircuit[Toffoli[S[1], S[2], S[3]]]  
toff = ExpressionFor[qc]
```



$$\frac{3}{4} - \frac{1}{4} S_1^z S_2^z - \frac{1}{4} S_1^z S_3^x - \frac{1}{4} S_2^z S_3^x + \frac{1}{4} S_1^z S_2^z S_3^x + \frac{S_1^z}{4} + \frac{S_2^z}{4} + \frac{S_3^x}{4}$$

It can be implemented by a combination of two-qubit gates.

```
In[7]:= gray = GrayControlledU[S@{1, 2}, S[3, 1]];
qc = QuantumCircuit[gray];
expr = ExpressionFor[qc];
toff - expr
```

```
Out[7]=
```

$$\frac{3}{4} - \frac{1}{4} S_1^z S_2^z - \frac{1}{4} S_1^z S_3^x - \frac{1}{4} S_2^z S_3^x + \frac{1}{4} S_1^z S_2^z S_3^x + \frac{S_1^z}{4} + \frac{S_2^z}{4} + \frac{S_3^x}{4}$$

```
Out[7]= 0
```

Here $V = \sqrt{X}$, and its matrix representation looks like this.

```
In[8]:= matV = MatrixPower[ThePauli[1], 1/2];
matV * 2 / (1 + I) // MatrixForm
opV = Elaborate@ExpressionFor[matV, S[3]]
```

```
Out[8]//MatrixForm=
```

$$\begin{pmatrix} 1 & -i \\ -i & 1 \end{pmatrix}$$

```
Out[8]= \left(\frac{1}{2} + \frac{i}{2}\right) + \left(\frac{1}{2} - \frac{i}{2}\right) S_3^x
```

Reversing the above circuit gives the identical result.

```
In[9]:= qc = QuantumCircuit[Reverse@gray];
expr = ExpressionFor[qc];
toff - expr
```

```
Out[9]=
```

$$\frac{3}{4} - \frac{1}{4} S_1^z S_2^z - \frac{1}{4} S_1^z S_3^x - \frac{1}{4} S_2^z S_3^x + \frac{1}{4} S_1^z S_2^z S_3^x + \frac{S_1^z}{4} + \frac{S_2^z}{4} + \frac{S_3^x}{4}$$

```
Out[9]= 0
```

As noted by Smolin & DiVincenzo (1996), it can also be reordered as following. This rearrangement is useful in optimizing the *Fredkin gate*, another universal gate for classical reversible computation.

This shows another slightly different implementation of the Toffoli gate.

```
In[1]:= qc = QuantumCircuit[Permute[gray, Cycles[{{4, 3, 2, 1}}]]];
expr = ExpressionFor[qc];
toff - expr
```

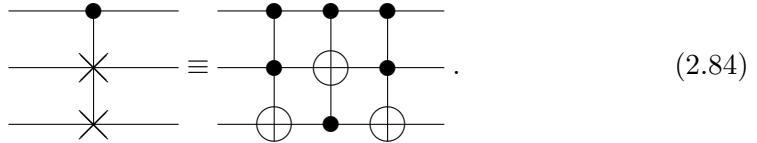
Out[1]=

$$\text{Out[1]} = \frac{3}{4} - \frac{1}{4} S_1^z S_2^z - \frac{1}{4} S_1^z S_3^x - \frac{1}{4} S_2^z S_3^x + \frac{1}{4} S_1^z S_2^z S_3^x + \frac{S_1^z}{4} + \frac{S_2^z}{4} + \frac{S_3^x}{4}$$

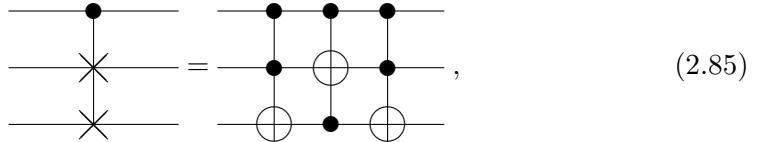
Out[1]=

$$\Theta$$

The Fredkin gate “swaps” the states of the two target qubits when the control qubit is set in the state $|1\rangle$, as depicted by the following two equivalent quantum circuit elements



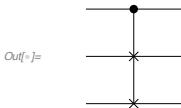
In fact, the relation between the SWAP gate and the CNOT gate suggests that there is another equivalent quantum circuit model of the Fredkin gate,



which is simpler than the above. This equivalent quantum circuit model was used by [Smolin & DiVincenzo \(1996\)](#) for an efficient implement of the Fredkin gate in terms of the elementary gates: Like the Toffoli gate, the Fredkin gate is not universal for quantum computation.

This shows the quantum circuit model of the quantum Fredkin gate.

```
In[1]:= qc = QuantumCircuit[Fredkin[S[1], S[2], S[3]]]
```



This is an explicit operator expression of the Fredkin gate in terms of the Pauli operators.

```
In[1]:= op = Elaborate[qc]
```

$$\frac{3}{4} + \frac{1}{4} S_2^x S_3^x + \frac{1}{4} S_2^y S_3^y + \frac{1}{4} S_2^z S_3^z - \frac{1}{4} S_1^z S_2^x S_3^x - \frac{1}{4} S_1^z S_2^y S_3^y - \frac{1}{4} S_1^z S_2^z S_3^z + \frac{S_1^z}{4}$$

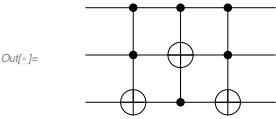
This is the matrix representation of the Fredkin gate in the logical basis.

```
In[7]:= mat = Matrix[qc];
mat // MatrixForm
```

$$\begin{pmatrix} 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \end{pmatrix}$$

The Fredkin gate is equivalent to a combination of three Toffoli gates.

```
In[8]:= qc2 = QuantumCircuit[
  Toffoli[S[1], S[2], S[3]],
  Toffoli[S[1], S[3], S[2]],
  Toffoli[S[1], S[2], S[3]]]
```



```
In[9]:= op2 = Elaborate[qc2]
```

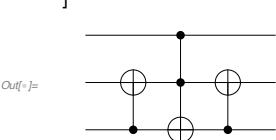
$$\frac{3}{4} + \frac{1}{4} S_2^x S_3^x + \frac{1}{4} S_2^y S_3^y + \frac{1}{4} S_2^z S_3^z - \frac{1}{4} S_1^z S_2^x S_3^x - \frac{1}{4} S_1^z S_2^y S_3^y - \frac{1}{4} S_1^z S_2^z S_3^z + \frac{S_1^z}{4}$$

```
In[10]:= op - op2
```

$$\text{Out[10]} = 0$$

In fact, the relation between the SWAP gate and the CNOT gate suggests that there is another equivalent quantum circuit model of the Fredkin gate.

```
In[11]:= new = QuantumCircuit[
  CNOT[S[3], S[2]],
  Toffoli[S[1], S[2], S[3]],
  CNOT[S[3], S[2]]]
```



```
In[12]:= qc - new // Elaborate
```

$$\text{Out[12]} = 0$$

2.4 Universal Quantum Computation

In classical computation, it is known that a finite set of logic gates—typically including AND, OR, and NOT—is sufficient to calculate any binary function. The set is said to be *universal* for classical computation. One can ask whether there exists a universal set of elementary quantum logic gates for quantum computation that enables to implement any arbitrary quantum unitary operation? In this section, we examine this question.

Consider a system of n qubits. Suppose that we want to implement an arbitrary unitary operation \hat{U} on the n -qubit register. We start by decomposing \hat{U} into a set of two-level unitary transformations, which we discussed for the case of two-qubit systems in Section 2.2.3.

Consider a three-qubit system, and an arbitrary unitary operation on it.

```
In[1]:= $n = 3;
SS = S[Range[$n], None];
mat = RandomUnitary[2^n];
mat // ; 3, ; 3] // MatrixForm
```

Out[1]=

$$\begin{pmatrix} -0.11099 + 0.146854 i & 0.577172 + 0.0870057 i & 0.244982 + 0.191715 i \\ -0.603563 + 0.147558 i & 0.00689927 - 0.0319512 i & -0.353264 + 0.00649725 i \\ -0.220451 - 0.257014 i & -0.239578 + 0.163395 i & 0.0478559 - 0.073729 i \end{pmatrix}$$

```
In[2]:= op = ExpressionFor[mat, SS];
qc = QuantumCircuit[{op, "Label" → "U"}]
```

Out[2]=

```
In[3]:= twl = TwoLevelDecomposition[mat];
MatrixForm /@ Matrix /@ twl // ; 3] // TableForm
```

Out[3]=

$$\begin{array}{c} \left(\begin{array}{cccccc|cc} 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1. + 0. i & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0.263136 + 0.964759 i \end{array} \right) \\ \left(\begin{array}{cccccc|cc} 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & -0.0435169 + 0.691903 i & -0.718129 - 0.0605535 i \\ 0 & 0 & 0 & 0 & 0 & 0 & 0.718129 - 0.0605535 i & -0.0435169 - 0.691903 i \end{array} \right) \\ \left(\begin{array}{cccccc|cc} 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0.40679 - 0.346824 i & 0.845124 \\ 0 & 0 & 0 & 0 & 0 & -0.845124 & 0.40679 + 0.346824 i & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \end{array} \right) \end{array}$$

Now a remaining question is how to implement the two-level unitary transformation in terms of elementary gates.

Let us consider a particular two-level unitary matrix on a three-qubit system. We want to implement it in terms of elementary quantum logic gates.

```
In[=]:= x = 4; y = 5;
U = TheRotation[Pi / 3, 2];
mat = Matrix@TwoLevelU[U, {x, y}, 2^n];
mat // MatrixForm
```

$$\text{Out}[=]/\text{MatrixForm}= \begin{pmatrix} 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & \frac{-\sqrt{3}}{2} & -\frac{1}{2} & 0 & 0 & 0 \\ 0 & 0 & 0 & \frac{1}{2} & \frac{\sqrt{3}}{2} & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \end{pmatrix}$$


```
In[=]:= op = ExpressionFor[mat, SS]
Out[=]= \frac{1}{8} \times (6 + \sqrt{3}) + \frac{1}{8} \times (2 - \sqrt{3}) S_1^z S_2^z +
\frac{1}{8} \times (2 - \sqrt{3}) S_1^z S_3^z + \frac{1}{8} \times (-2 + \sqrt{3}) S_2^z S_3^z - \frac{1}{2} S_1^+ S_2^- S_3^- + \frac{1}{2} S_1^- S_2^+ S_3^+

```

Our implementation is based on the Gray code sequence. Notice the function Reverse.

```
In[=]:= gates = Flatten@GrayTwoLevelU[U, {x, y}, SS]
Out[=]= \{S_1^x, \text{CNOT}[\{S_1, S_2\}, \{S_3\}], S_1^x, S_3^x, \text{CNOT}[\{S_2, S_3\}, \{S_1\}], S_3^x, \text{CNOT}[\{S_1, S_2\}, \{S_3\}], \text{CNOT}[\{S_1, S_3\}, \{S_2\}], S_2^x, \text{ControlledU}[\{S_1, S_2\}, \frac{\sqrt{3}}{2} + \frac{i}{2} S_3^y, \text{Label} \rightarrow U], S_2^x, \text{CNOT}[\{S_1, S_3\}, \{S_2\}], \text{CNOT}[\{S_1, S_2\}, \{S_3\}], S_3^x, \text{CNOT}[\{S_2, S_3\}, \{S_1\}], S_3^x, S_1^x, \text{CNOT}[\{S_1, S_2\}, \{S_3\}], S_1^x\}
```

```
In[=]:= new = Apply[Dot, Matrix[#, SS] & /@ Elaborate /@ gates] // Simplify;
new // MatrixForm
```

$$\text{Out}[=]/\text{MatrixForm}= \begin{pmatrix} 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & \frac{-\sqrt{3}}{2} & -\frac{1}{2} & 0 & 0 & 0 \\ 0 & 0 & 0 & \frac{1}{2} & \frac{\sqrt{3}}{2} & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \end{pmatrix}$$


```
In[=]:= qc1 = QuantumCircuit[gates[[;; 10]]];
qc2 = QuantumCircuit[gates[[11 ;;]]]
```

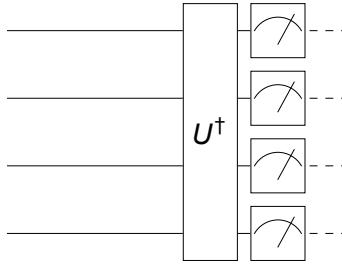


Figure 2.6: Measurement in a basis $\{|\alpha_x\rangle\}$ other than the logical basis. The unitary operator \hat{U} here corresponds to the basis change $|\alpha_y\rangle = \hat{U} |y\rangle = \sum_x |x\rangle \langle x| \alpha_y\rangle$.

```
In[7]:= new = Matrix[qc2].Matrix[qc1] // Simplify;
new // MatrixForm
Out[7]//MatrixForm=
```

$$\begin{pmatrix} 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & \frac{\sqrt{3}}{2} & -\frac{1}{2} & 0 & 0 & 0 \\ 0 & 0 & 0 & \frac{1}{2} & \frac{\sqrt{3}}{2} & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \end{pmatrix}$$

So far, we have seen that one can implement any unitary operation combining the CNOT and single-qubit gates. A subtle issue is that single-qubit gates form a continuum of unitary operators. It is impractical to implement them to an infinite accuracy. Fortunately, it is known that a discrete set of gates is sufficient to perform universal quantum computation: The Hadamard, quadrant phase, octant phase, and CNOT gates form a universal set of gates in the sense that an arbitrary gate on n qubits can be approximated to an arbitrary accuracy using a quantum circuit model composed of only these gates. The set remains universal if octact phase gates are replaced with Toffoli gates.

2.5 Measurements

We conclude this chapter with a few discussions on measurement. In quantum computers, measurement is assumed to be performed independently on individual qubits in the logical basis, $\{|x\rangle : x = 0, 1, \dots, 2^n - 1\}$.

What if a measurement in another basis, say, $\{|\alpha_x\rangle = \hat{U} |x\rangle\}$ is required? We require that the input state $|\alpha_x\rangle$ should end up with the logical basis state $|x\rangle$ with unit probability so that the measurement yields the outcome x . This process is described by the measurement operators $\hat{M}_x := |x\rangle \langle \alpha_x| = |x\rangle \langle x| \hat{U}^\dagger$. Evidently they satisfy the condition $\sum_x \hat{M}_x^\dagger \hat{M}_x = \hat{I}$ for measurement operators (see Postulate 1.3). Now we note that the operators $\hat{P}_x := |x\rangle \langle x|$ describe nothing but

the measurement in the logical basis. It implies that simply by applying the inverse unitary operation \hat{U}^\dagger before the measurement, the measurement in the new basis $\{| \alpha_x \rangle\}$ can be achieved through the measurement in the logical basis; see Fig. 2.6. For example, suppose that a qubit is in the state $|\psi\rangle = |0\rangle c_0 + |1\rangle c_1$. By default, a measurement is assumed to be in the logical basis and the measurement statistics reflects the probability distribution $P_0 = |c_0|^2$ and $P_1 = |c_1|^2$ as illustrated in the demonstration below.

First consider a measurement in the logical basis.

```
In[1]:= Let[Complex, c]
vec = ProductState[S[1] → {c[0], c[1]}];
qc = QuantumCircuit[vec, "Spacer", Measurement[S[1]], "PortSize" → {2, 0.2}]
Out[1]= |0⟩c₀ + |1⟩c₁ ─── [Measure] ---.
```

```
In[2]:= Block[
{c, data},
c[0] = 1/2;
c[1] = Sqrt[3]/2;
data = Table[out = ExpressionFor[qc]; Readout[out, S[1]], 1000];
Histogram[data, FrameLabel → {"Readout", "Counts"}, ImageSize → Small]
]
Out[2]=
```

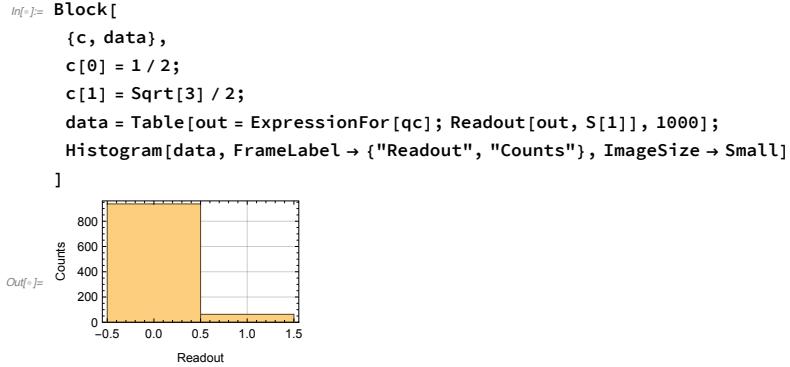
We now want to make a measurement on the eigenbasis $\{| \pm \rangle\}$ of the Pauli X operator. In this case, it is the Hadamard gate that gives the desired basis change. In the new basis, the state vector $|\psi\rangle$ is given by

$$|\psi\rangle = |+\rangle \frac{c_0 + c_1}{\sqrt{2}} + |-\rangle \frac{c_0 - c_1}{\sqrt{2}}, \quad (2.86)$$

Now the measurement statistics is in accordance with the probability distribution $P_{\pm} = |c_0 \pm c_1|^2/2$.

Now consider a measurement in the eigen-basis of the Pauli X operator.

```
In[3]:= Let[Complex, c]
vec = ProductState[S[1] → {c[0], c[1]}];
qc = QuantumCircuit[vec, S[1, 6], Measurement[S[1]], "PortSize" → {2, 0.2}]
Out[3]= |0⟩c₀ + |1⟩c₁ ─── [H] [Measure] ---.
```

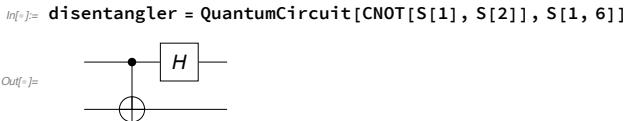


Another interesting example is the *Bell measurement*. Bell measurement is a measurement on two qubits in the basis of Bell states,

$$\begin{aligned} |\beta_0\rangle &= \frac{|00\rangle + |11\rangle}{\sqrt{2}} \\ |\beta_1\rangle &= \frac{|01\rangle + |10\rangle}{\sqrt{2}} \\ |\beta_2\rangle &= \frac{|01\rangle - |10\rangle}{\sqrt{2}} \\ |\beta_3\rangle &= \frac{|00\rangle - |11\rangle}{\sqrt{2}} \end{aligned} \quad (2.87)$$

Recall that the Bell states can be generated from the logical basis states by the so-called quantum entangler circuit, a combination of the Hadamard gate and the CNOT gate discussed in Section 2.2.1. Therefore, the Bell measurement can be achieved by applying the inverse of the entangling operation.

This is the "disentangler" quantum circuit, which is the inverse of the entangler quantum circuit



This shows that the disentangler quantum circuit maps the Bell states into the logical basis states.

In[=]:= **op** = ExpressionFor[disentangler];
bs = BellState@S@{1, 2};
Thread[bs -> op ** bs] // LogicalForm // TableForm

Out[=]:= TableForm[

$\frac{ \theta_{S_1}\theta_{S_2}\rangle + 1_{S_1}1_{S_2}\rangle}{\sqrt{2}}$	$\rightarrow \theta_{S_1}\theta_{S_2}\rangle$
$\frac{ \theta_{S_1}1_{S_2}\rangle + 1_{S_1}\theta_{S_2}\rangle}{\sqrt{2}}$	$\rightarrow \theta_{S_1}1_{S_2}\rangle$
$\frac{ \theta_{S_1}1_{S_2}\rangle - 1_{S_1}\theta_{S_2}\rangle}{\sqrt{2}}$	$\rightarrow 1_{S_1}1_{S_2}\rangle$
$\frac{ \theta_{S_1}\theta_{S_2}\rangle - 1_{S_1}1_{S_2}\rangle}{\sqrt{2}}$	$\rightarrow 1_{S_1}\theta_{S_2}\rangle$

For fast quantum algorithms and quantum error corrections, more sophisticated measurements may be necessary. A common example is the quantum phase estimation, which is one of the core parts of Shor’s factorization algorithm. We will discuss it in Section 4.4.

Problems

- 2.1. Let $\hat{\Phi}(\phi)$ be the *phase gate*, which gives rise to a *relative* phase shift by $\phi \in [0, 2\pi)$,

$$\hat{\Phi}(\phi) : |0\rangle \mapsto |0\rangle, \quad |1\rangle \mapsto |1\rangle e^{i\phi}. \quad (2.88)$$

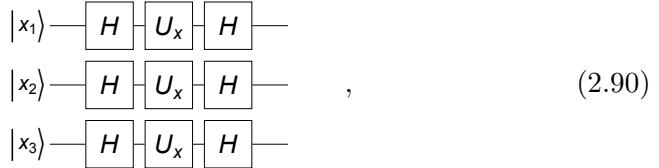
- (a) Show that on a n -qubit quantum register,

$$[\hat{\Phi}(\phi)]^{\otimes n} |x\rangle = |x\rangle e^{i\phi(x_1 + \dots + x_n)} \quad (2.89)$$

for any $x = 0, 1, \dots, 2^n - 1$.

- (b) Evaluate explicitly the state $[\hat{\Phi}(\phi)]^{\otimes n} \hat{H}^{\otimes n} |0\rangle$, where \hat{H} is the Hadamard gate.

- 2.2. Let $\hat{U}_x(\phi)$ be the rotation around the x -axis by angle ϕ on a single qubit. Analyze and evaluate explicitly the following quantum circuit model



where $|x\rangle := |x_1\rangle \otimes |x_2\rangle \otimes |x_3\rangle$ is a 3-qubit logical basis state and the labels “ H ” and “ U_x ” indicate the Hadamard gate \hat{H} and the rotation gate $\hat{U}_x(\phi)$, respectively. Generalize the result and show that

$$\hat{H}^{\otimes n} [\hat{U}_x(\phi)]^{\otimes n} \hat{H}^{\otimes n} |x\rangle = e^{-i\phi n/2} |x\rangle e^{i\phi(x_1 + \dots + x_n)} \quad (2.91)$$

for any $x = 0, 1, \dots, 2^n - 1$.

- 2.3. Let \hat{S}^μ be the Pauli operators. Show that

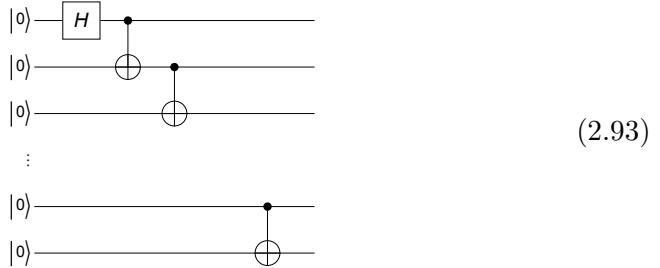
$$e^{i\hat{S}^\mu \phi/2} \hat{S}^\nu e^{-i\hat{S}^\mu \phi/2} = \hat{S}^\nu \cos(\phi) + \hat{S}^\lambda \epsilon_{\lambda\mu\nu} \sin(\phi) \quad (2.92)$$

for all $\mu, \nu = x, y, z$ and $\mu \neq \nu$.

- 2.4. The lever on Bob’s quantum computer is stuck in the “forward” position, so it can only perform CNOT gates controlled by qubit 1 on qubit 2 (target). His computer can still perform single-qubit operations normally. How can he perform a CNOT gate controlled by qubit 2 on qubit 1?^{2.3}

^{2.3}This problem was published in [Gottesman \(1999\)](#).

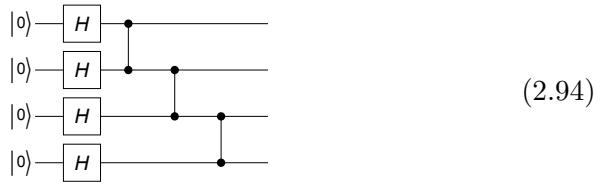
2.5. Consider the following quantum circuit model with n qubits:



Show that it generates the Greenberg-Horne-Zeilinger state in Eq. (2.39), identical to the one from the quantum circuit model in Fig. 2.5 (b).

2.6. Consider a quantum register of four qubits.

(a) Analyze the following quantum circuit model



and evaluate the resulting state $|\Psi\rangle$ explicitly. The state is a so-called *cluster state* or *graph state*, a crucial resource in the measurement-based quantum computation—see Section 3.4.

(b) Show that in the state $|\Psi\rangle$ from (b), every qubit is *maximally entangled* with the rest qubits, that is, the *reduced density matrix* $\hat{\rho}_j$ of the j th qubit, $\hat{\rho}_j := \text{Tr}_{k \neq j} |\Psi\rangle \langle \Psi|$, is given by $\hat{\rho}_j = \hat{I}/2$ —it exhibits coherence in no basis.

2.7. Suppose that a qubit is known to be in one of the two eigenstates of the unitary operator

$$\hat{U} = \hat{\sigma}^0 \cos(\phi/2) - i\hat{\sigma}^x \sin(\phi/2) \quad (2.95)$$

with the angle ϕ known. Construct a quantum circuit model to figure out the unknown state using an additional qubit.

Hint: Use a controlled- U gate to acquire the one-bit information. This is a simplified version of the quantum phase estimation procedure (see Section 4.4), but it can be worked out without resorting to it.

2.8. Suppose that a two-qubit system is known to be in one of the four eigenstates of the unitary operator

$$\hat{U} = e^{i\phi} (|0\rangle \langle 0| + i|1\rangle \langle 1| - |2\rangle \langle 2| - i|3\rangle \langle 3|). \quad (2.96)$$

Construct a quantum circuit model to figure out the unknown state using two additional qubits.

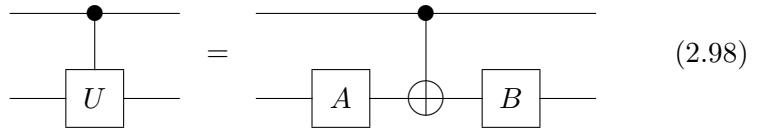
Hint: Use the property (2.18) of the Hadamard gate and those of the controlled- U gates, where a unitary operator acts on a two-qubit system controlled by a single qubit.

- 2.9. Consider the following quantum circuit model consisting of the CNOT gates on a n -qubit quantum register



- (a) Find the output state for the input of a logical basis state $|x_1\rangle \otimes \cdots \otimes |x_n\rangle \in \mathcal{S}^{\otimes n}$.
 - (b) Find the output state for the input state $|+\rangle \otimes \cdots \otimes |+\rangle \otimes |-\rangle$, where $|\pm\rangle := (|0\rangle \pm |1\rangle)/\sqrt{2}$.
- 2.10. Let \hat{U} be a unitary operator on a single qubit. Show that the following three statements are equivalent:

- (a) There exist unitary gates \hat{A} and \hat{B} such that



- (b) There exists a unitary operator \hat{W} such that $\hat{U} = \hat{W}\hat{Z}\hat{W}^\dagger$.
- (c) $\text{Tr } \hat{U} = 0$ and $\det \hat{U} = -1$.

- 2.11. Let $P(i \leftrightarrow j)$ be the matrix exchanging the i th and j th rows (columns) of vectors/matrices. For example, on a four-dimensional space,

$$P(1 \leftrightarrow 2) = \begin{bmatrix} 0 & 1 & & \\ 1 & 0 & & \\ & & 1 & \\ & & & 1 \end{bmatrix}. \quad (2.99)$$

$\hat{P}(i \leftrightarrow j)$ is the corresponding operator.

- (a) Find the quantum circuit model for $\hat{P}(2 \leftrightarrow 3)$ on a two-qubit system (four-dimensional vector space $\mathcal{S} \otimes \mathcal{S}$) using CNOT gates only.

- (b) Find the quantum circuit model for $\hat{P}(2 \leftrightarrow 3)$ on a three-qubit system ($\mathcal{S}^{\otimes 3}$) using only multi-qubit CNOT gates only.
- (c) Find the quantum circuit model for $\hat{P}(4 \leftrightarrow 7)$ on a three-qubit system ($\mathcal{S}^{\otimes 3}$) using only multi-qubit CNOT gates onlys.
- 2.12. Let $\hat{U} = e^{i\phi} \hat{I}$ be a single-qubit unitary operator that *globally* shifts the phase by ϕ . Also define

$$\hat{T} = |0\rangle\langle 0| + e^{i\phi} |1\rangle\langle 1| \doteq \begin{bmatrix} 1 & 0 \\ 0 & e^{i\phi} \end{bmatrix}, \quad (2.100)$$

which shifts the *relative* phase by ϕ .

- (a) Show that

$$\text{Diagram: } \begin{array}{c} \bullet \\ \hline \square U \end{array} = \begin{array}{c} \square T \\ \hline \end{array}, \quad (2.101)$$

where the labels “ U ” and “ T ” denote the unitary operators \hat{U} and \hat{T} , respectively.

- (b) Show also that

$$\text{Diagram: } \begin{array}{c} \bullet \\ \hline \bullet \\ \hline \bullet \\ \hline \square U \end{array} = \begin{array}{c} \square T \\ \hline \bullet \\ \hline \bullet \\ \hline \end{array} = \begin{array}{c} \bullet \\ \hline \square T \\ \hline \bullet \\ \hline \bullet \\ \hline \end{array} = \begin{array}{c} \bullet \\ \hline \bullet \\ \hline \square T \\ \hline \end{array}. \quad (2.102)$$

Chapter 3

Virtual Realizations of Quantum Computers

- August 6, 2021 (v1.17)

In the previous chapter, we have seen how quantum computation works under the assumption that elementary quantum logic gates are available. But how can one build a quantum computer, a machine, that allows such quantum logic gates? Quantum computers are physical systems and the implementation of all quantum logic gates are governed by the laws of physics. In this chapter, we discuss the basic physical principles that are directly involved in the implementation of quantum logic gates. Through the course of the discussion, we will find some basic conditions and requirements that one has to fulfill to build a quantum computer.

By now there are many quantum computer architectures that are not only proposed and tested at the research level but also actually running. However, understanding each architecture requires a certain level of knowledge about the physical systems. For example, to understand a quantum computer based on superconducting circuits, one has to first understand the superconductivity, the Josephson effect, the flux quantization, the Josephson inductance (a sort of non-linear kinetic inductance), and the interaction of superconducting circuits to electromagnetic fields. Such discussions often hinder access to the essential part of the operating principle of a quantum computer, and are out of the scope of this workbook.

Here we consider an idealistic and minimal quantum system that is suitable for quantum computation and discuss how to control it to implement the desired quantum logic gates. It is certainly impractical. Nevertheless, it will point out the key requirements when one wants to actually develop a quantum computer based on realistic systems and devices. Through the discussions, we will indicate how the relevant parts are related to actual quantum computer architectures.



Figure 3.1: IBQ quantum computer based on superconducting circuits.

3.1 Quantum Bits

We have already noted several times that the building blocks and basic computational units of a quantum computer are qubits. Ideally, a qubit is associated with a two-dimensional Hilbert space. In reality, the Hilbert space for any realistic system is infinite-dimensional, and a qubit usually refers to *certain degrees of freedom* that are relatively independent of other degrees of freedom. For example, the spin of an electron or the polarization of a photon has exactly two-dimensional Hilbert space. In many cases, a qubit may also refer to a *certain two-dimensional subspace* of a larger Hilbert space that is decoupled or separated relatively well from the rest. For example, a superconducting charge qubit refers to the two lowest-energy charging states in a small—hundreds of nanometers in lateral size—superconducting island.

However, a well-defined two-dimensional Hilbert space (or subspace) does not necessarily mean that the degrees of freedom in question qualifies as a qubit. For example, consider the spin of a neutron. Although its Hilbert space dimension is certainly two, you recognize that it can hardly be used for quantum computation. It is hard to isolate a neutron, and even more so to manipulate its spin in a reliable and tunable manner. Then what requirements should qubits—individually and as a whole collection—meet to build a practical quantum computer? Apart from specific technical issues in specific systems, these are the basic requirements—the so-called DiVincenzo criteria—commonly rated to assess the potential of a particular architecture in consideration ([DiVincenzo, 2000](#)):

- (a) The qubits should be well characterized and form a scalable system. For each qubit, the Hilbert space should be well defined in the sense mentioned above and its internal Hamiltonian including the parameters needs to be accurately known. The qubits must also admit genuine interactions among them and maintain the characteristics up to a sufficiently large scale for practical computation.

- (b) The qubits should allow initialization to a fixed logical basis state. Even though any quantum algorithm assumes superposition in the middle of the process, all computations must start from a known value. This straightforward requirement is the same even for classical computers. One of the common approaches for initialization is to cool down the system and wait for it to relax to the ground state. Another method is to perform a projective measurement in the logical basis so as for the state to collapse to the logical basis state corresponding to the measurement outcome.
- (c) The qubits should maintain coherence long enough for the desired gate operations. The superposition between different logical basis states is a crucial difference distinguishing quantum computers from classical computers. Unfortunately, qubits are subject to various decoherence effects due to external control circuits and measuring devices and eventually lose quantumness. The system should maintain the coherence during the desired gate operations to get a reliable result out of the computation.
- (d) The system of qubits should allow a *universal* set of quantum gate operations. As discussed in Chapter 2, quantum computation is to achieve a desired unitary transformation with a combination of certain elementary gate operations that are acting on a single qubit or two qubits at a time. Below we will discuss the physical implementations of those elementary quantum logic gates.
- (e) The system should allow measurements in the logical basis. At the end of a computation, the result needs to be read out, and it is achieved by performing measurements on specific qubits. The capability of accurate measurement is called the *quantum efficiency*. Ideal measurement has 100% quantum efficiency. Less than 100 % quantum efficiency in measurements leads to a tradeoff with other resources. For example, if a computation is desired with 97 % reliability while measurements have only 90 % quantum efficiency, then one must repeat the computation three times or more.

In the rest of the chapter, let us now focus on the manipulation of quantum states of qubits, which naturally forms the largest part of quantum computation.

Consider a quantum computer consisting of n qubits. Let \mathcal{S}_j ($j = 1, \dots, n$) be the 2-dimentional Hilbert space associated with the j th single qubit. An ideal quantum computer would realize a Hamiltonian on $\mathcal{S}_1 \otimes \dots \otimes \mathcal{S}_n$ of the form

$$\hat{H}(t) = \sum_j \sum_{\mu} B_j^{\mu}(t) \hat{S}_j^{\mu} + \sum_{ij} \sum_{\mu\nu} J_{ij}^{\mu\nu}(t) \hat{S}_i^{\mu} \hat{S}_j^{\nu}, \quad (3.1)$$

where \hat{S}_j^{μ} ($\mu = x, y, z$) are the Pauli operators—see Section 2.1.1—on \mathcal{S}_j .

The parameter B_j^{μ} directly controls the j th qubit. Physically, it plays the same role as the magnetic field on a spin. In realistic systems, it may be hard to address

single qubits individually. How freely one can manipulate single qubits strongly depends on how many of the parameters B_j^μ the system allows to tune accurately. See, for example, Section 3.2 below.

The control parameters $J_{ij}^{\mu\nu}$ describe the (hypothetical) exchange coupling between the i th and j th qubits. In principle, any type of interaction between two qubits can be used to implement CNOT gate (see Section 3.2 for examples) although the actual implementation may require more than one interactions and many additional single-qubit operations depending on the particular type of coupling. Therefore, an accurate control of the coupling parameters $J_{ij}^{\mu\nu}$ between a specific pair of qubits is essential for universal quantum computation. In realistic systems, the coupling parameters $J_{ij}^{\mu\nu}$ are even more difficult to realize. First of all, in many architectures the connectivity of qubits that are not in direct proximity of each other is seriously limited. Further, dynamically turning on and off the coupling is often forbidden. In many cases, in order to achieve a sizable strength, the exchange couplings are kept turned on throughout the whole computation processes. Such difficulties and imperfections all contribute to the errors in the computational outputs.

We will be denoting each qubit by the symbol S and accompanying indices .

Let[Qubit, S]

The Pauli operators are specified by the last index. For example, the Pauli operator S_j^x is denoted by $S[j,1]$.

```
In[1]:= S[j, 1]
Out[1]= Sxj

In[2]:= S[j, 1] ** S[j, 2]
          S[j, 1] ** S[k, 2]
Out[2]= i Szj

Out[3]= Sxj Syk
```

3.2 Dynamical Scheme

The dynamic scheme implements the desired quantum gates through the time-evolution operator governing the dynamics of the physical qubits in a quantum computer; hence the name. It is the most common scheme of quantum computation. A majority of quantum computers demonstrated so far are based on this scheme. In this section, we briefly survey the elementary principles and common methods to employ to achieve elementary single-qubit and two-qubit quantum gates in the dynamical scheme.

3.2.1 Implementation of Single-Qubit Gates

Conceptually, the most straightforward way to control a single qubit is to apply the static parameters B^x , B^y , and B^z for a certain period τ of operation time. We refer to them collectively by a vector $\mathbf{B} = (B^x, B^y, B^z)$. One can regard it as a fictitious magnetic field—but the dimension of \mathbf{B} is energy unlike the real magnetic field. The Hamiltonian for the qubit is given by

$$\hat{H} = \frac{1}{2} \mathbf{B} \cdot \hat{\mathbf{S}}, \quad (3.2)$$

where $\hat{\mathbf{S}} := (\hat{S}^x, \hat{S}^y, \hat{S}^z)$ is the vector of the Pauli operators on \mathcal{S} . The time evolution is then governed by the unitary operator

$$\hat{U}(t) = \exp(-it\hat{H}) = \exp(-it\mathbf{B} \cdot \hat{\mathbf{S}}/2) \quad (3.3)$$

in accordance with Postulate 2. It has the same form as the single-qubit rotation operator in Eq. (2.23), Section 2.1.3: It describes the rotation in the Pauli space—or the Bloch sphere—around the axis parallel to the vector \mathbf{B} by the angle $\phi(t) = |\mathbf{B}|t$. When the involved two-level system is a true spin, such a rotation in the Pauli space is called the *Larmor precession*.

Consider a qubit. Let us apply a (fictitious) magnetic field.

```
In[1]:= Let[Real, B]
opH = Dot[B@{1, 2, 3}, S@{1, 2, 3}] / 2
Out[1]=  $\frac{1}{2} (B_1 S^x + B_2 S^y + B_3 S^z)$ 
```

To be specific, we consider the case with the magnetic field between the x- and z-axis in the xz-plane. The factor of $1/\sqrt{2}$ is to set the energy (frequency) scale to unit.

```
In[2]:= B[1] = B[3] = 1/Sqrt[2]; B[2] = 0;
opH
```

```
Out[2]=  $\frac{1}{2} \left( \frac{S^x}{\sqrt{2}} + \frac{S^z}{\sqrt{2}} \right)$ 
```

```
In[3]:= opU[t_] = MultiplyExp[-I t opH]
opU[t] // Elaborate
```

```
Out[3]=  $e^{-\frac{1}{2} I t \left( \frac{S^x}{\sqrt{2}} + \frac{S^z}{\sqrt{2}} \right)}$ 
```

```
Out[3]=  $\cos\left[\frac{t}{2}\right] - \frac{i S^x \sin\left[\frac{t}{2}\right]}{\sqrt{2}} - \frac{i S^z \sin\left[\frac{t}{2}\right]}{\sqrt{2}}$ 
```

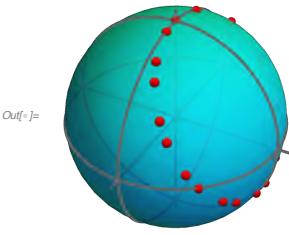
```
In[4]:= vec[t_] = opU[t] ** Ket[] // Elaborate
```

```
Out[4]=  $\frac{i |1s\rangle \sin\left[\frac{t}{2}\right]}{\sqrt{2}} + |-\rangle \left( \cos\left[\frac{t}{2}\right] - \frac{i \sin\left[\frac{t}{2}\right]}{\sqrt{2}} \right)$ 
```

This illustrates the Larmor precession (with the qubit regarded as a “spin”).

Technical Note: You need Chop because numerical errors sometimes give $0.+Ket[...]$, which cannot be handled properly by BlochVector.

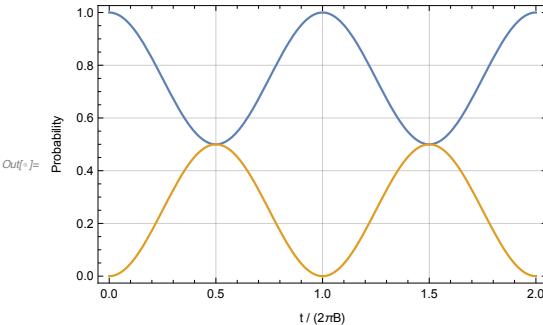
```
In[5]:= vv = BlochVector /@ Table[Chop@vec[t], {t, 0, 8, 0.5}];
BlochSphere[{Red, Bead /@ vv}, ImageSize -> Small]
```



Let us now examine the probabilities for the qubit to be found in the logical basis states.

```
In[6]:= prob[t_] = Abs[Normal@Matrix@vec[t]]^2
Out[6]= \left\{ \left| \cos\left[\frac{t}{2}\right] - \frac{i \sin\left[\frac{t}{2}\right]}{\sqrt{2}} \right|^2, \frac{1}{2} \left| \sin\left[\frac{t}{2}\right] \right|^2 \right\}
```

```
In[7]:= Plot[Evaluate@prob[2 Pi t], {t, 0, 2},
FrameLabel -> {"t / (2\pi B)", "Probability"}]
```



Although conceptionally simple, the above method has limited applications in many realistic systems. For example, in the presence of other levels, one cannot apply the method selectively between the two levels in question. More widely applicable method is to apply an oscillating field: Suppose that

$$B^x = B_\perp \cos(\omega t), \quad B^y = B_\perp \sin(\omega t), \quad B^z = B_0. \quad (3.4)$$

One can regaard it as a fictitious magnetic field precessing around the z -axis with frequency ω . The Hamiltonian now depends on time and is given by

$$\hat{H}(t) = \frac{1}{2}B_\perp [\cos(\omega t)\hat{S}^x + \sin(\omega t)\hat{S}^y] + \frac{1}{2}B_0\hat{S}^z. \quad (3.5)$$

Recalling the property in Eq. (2.25) of the Pauli operators as the generators of rotation, we observe that

$$\hat{U}_z^\dagger(\omega t)\hat{H}(t)\hat{U}_z(\omega t) = \frac{1}{2}B_\perp\hat{S}^x + \frac{1}{2}B_0\hat{S}^z \quad (3.6)$$

does not depend on time any longer. This observation suggests that the dynamics looks simpler in the rotating frame. As the rotating frame is not an inertial frame,

one has to take into account the non-inertial effect—corresponding to the classical *inertial force*: The state vectors $|\psi(t)\rangle$ and $|\psi_R(t)\rangle$ in the lab and rotating frame, respectively, are related by

$$|\psi_R(t)\rangle = \hat{U}_z^\dagger(\omega t) |\psi(t)\rangle . \quad (3.7)$$

Putting it into the original Schrödinger equation for $|\psi(t)\rangle$,

$$i\partial_t |\psi\rangle = \hat{H}(t) |\psi\rangle , \quad (3.8)$$

and operating $\hat{U}_z^\dagger(t)$ from the left on both sides, one can get the Schrödinger equation in the rotating frame,

$$i\partial_t |\psi_R\rangle = \hat{H}_R |\psi_R\rangle , \quad (3.9)$$

where the Hamiltonian in the rotating frame is given by

$$\hat{H}_R(t) := U_z^\dagger(\omega t) \hat{H}(t) \hat{U}_z(\omega t) - \hat{U}_z^\dagger(\omega t) [i\partial_t \hat{U}_z(\omega t)] . \quad (3.10)$$

The second term in \hat{H}_R is responsible for the non-inertial effect. As already expected in the above observation, the rotating-frame Hamiltonian does not depend on time any longer,

$$\hat{H}_R = \frac{1}{2} B_\perp \hat{S}^x + \frac{1}{2} (B_0 - \omega) \hat{S}^z . \quad (3.11)$$

The time-evolution operator in the rotating frame,

$$\hat{U}_R(t) = \exp(-it\hat{H}_R) \quad (3.12)$$

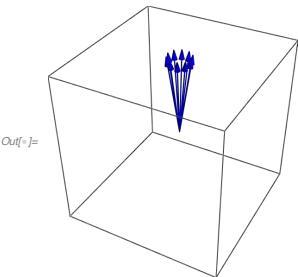
is *formally* the same as the one (3.3) in the lab frame. It describes a precession around the axis parallel to $\mathbf{B}_R := (B_\perp, 0, B_0 - \omega)$ with the frequency

$$\Omega_R := \sqrt{B_\perp^2 + (B_0 - \omega)^2} . \quad (3.13)$$

The precession in the rotating frame is called the *Rabi oscillation* and the frequency in (3.13) is called the *Rabi frequency*. The fictitious magnetic field in the rotating frame—compare Eqs. (3.2) and (3.10)—is almost along the x -axis for $\omega \approx B_0$, that is, the time-evolution $\hat{U}_R(t)$ in the rotating frame corresponds to the rotation around the x -axis in the Bloch sphere. In this case, the qubit can make a full transition from the initial state $|0\rangle$ to the orthogonal state $|1\rangle$ by properly tuning the operation time and/or the parameter B_\perp . In this sense, when $\omega \approx B_0$, the system is said to be at *resonance*. As the driving frequency ω gets away off the resonance, the maximum transition probability becomes smaller and smaller. This resonance behavior allows to induce transitions between a selected pair of two levels among many others.

Let us now apply a time-dependent field. It precesses around the z-axis. Note that typically, $B_3 \gg B_1, B_2$.

```
In[5]:= ω = 2 Pi;
B[1] = Cos[ω t];
B[2] = Sin[ω t];
B[3] = 1.1 ω; (* near resonance *)
Graphics3D[{Blue, Table[Arrow@Tube@{{0, 0, 0}, B@{1, 2, 3}}, {t, 0, 1, 0.1}]},
PlotRange → ω {{-1, 1}, {-1, 1}, {-1, 1}}, ImageSize → Small]
```



```
In[6]:= opH = Dot[B@{1, 2, 3}, S@{1, 2, 3}] / 2
Out[6]= 1/2 (Cos[2 π t] Sx + 6.9115 Sz + Sy Sin[2 π t])
```

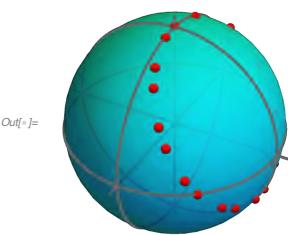
```
In[7]:= opHR =
Rotation[-ω t, S[3]] ** opH ** Rotation[ω t, S[3]] - ω S[3] / 2 // Elaborate // Chop
Out[7]= Sx/2 + 0.314159 Sz
```

```
In[8]:= opUR[t_] = MultiplyExp[-I t opHR]
Out[8]= e-i t (Sx/2 + 0.314159 Sz)
```

```
vec[t_] = opUR[t] ** Ket[] // Elaborate;
```

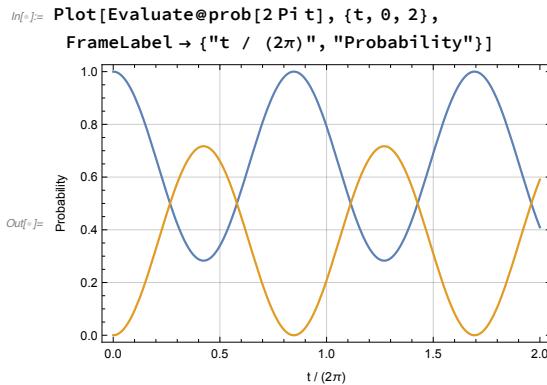
This illustrates the precession in the rotating frame, which is called the Rabi oscillation.

```
In[9]:= vv = BlochVector /@ Table[Chop@vec[t], {t, 0, 8, 0.5}];
BlochSphere[{Red, Bead /@ vv}, ImageSize → Small]
```



This shows the transition probabilities in the logical basis states. Note that the probabilities are the same both in the lab and rotating frame. The Rabi oscillation frequency is in this particular example is close to one---it is exactly one at resonance.

```
prob[t_] = Abs[Normal@Matrix@vec[t]]^2;
```



Let us consider the case exactly at the resonance.

```
w = 2 Pi;  
B[1] = Cos[w t];  
B[2] = Sin[w t];  
B[3] = w; (* resonance *)
```

In[6]:= opH = Dot[B@{1, 2, 3}, S@{1, 2, 3}] / 2
opHR =
Rotation[-w t, S[3]] ** opH ** Rotation[w t, S[3]] - w S[3] / 2 // Elaborate // Chop
opUR[t_] = MultiplyExp[-I t opHR]

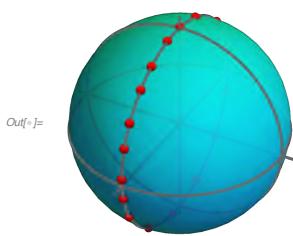
Out[6]= $\frac{1}{2} (\cos[2\pi t] S^x + 2\pi S^z + S^y \sin[2\pi t])$

Out[7]= $\frac{S^x}{2}$

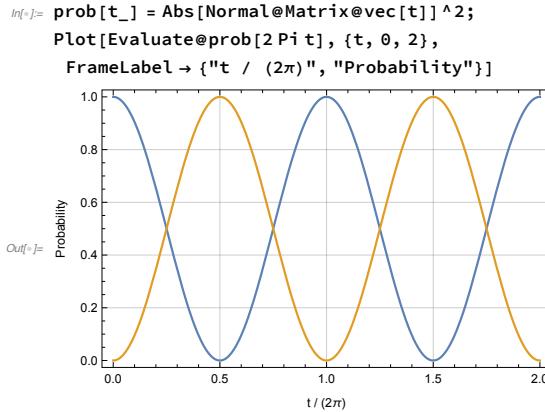
Out[8]= $e^{-\frac{1}{2} i t S^x}$

The Rabi oscillation now corresponds to the Larmor precession around the x-axis in the rotating frame.

In[9]:= vec[t_] = opUR[t] ** Ket[] // Elaborate;
vv = BlochVector /@ Table[Chop@vec[t], {t, 0, 8, 0.5}];
BlochSphere[{Red, Bead /@ vv}, ImageSize -> Small]



As the precession axis is exactly along the x-axis, the maximum transition probability can reach unity.



In the above discussion of Rabi oscillation, the two parameters are deliberately manipulated periodically with relative phase difference of $\pi/2$. It may not always be possible in practical experimental situations. Fortunately, however, the requirement is not so stringent: Suppose that only one parameter can be driven periodically, say, $B_y(t) = 2B_\perp \sin(\omega t)$ (notice the factor 2 introduced here for convenience). The time-dependent Hamiltonian

$$\hat{H}(t) = B_\perp \sin(\omega t) \hat{S}^y + \frac{1}{2} B_0 \hat{S}^z \quad (3.14)$$

looks seemingly simpler than the one in (3.5), but it does not allow for an exact solution. Instead, let us rewrite it into the form

$$\begin{aligned} \hat{H}(t) = & \frac{1}{2} B_0 \hat{S}^z + \frac{1}{2} B_\perp \left[\cos(\omega t) \hat{S}^x + \sin(\omega t) \hat{S}^y \right] \\ & - \frac{1}{2} B_\perp \left[\cos(-\omega t) \hat{S}^x + \sin(-\omega t) \hat{S}^y \right], \end{aligned} \quad (3.15)$$

which is obviously the same as (3.14). While the second term describes a fictitious magnetic field rotating in the counterclockwise direction, the field in the last term rotates in the clockwise direction. In the frame rotating in the counterclockwise sense, the Hamiltonian reads as

$$\hat{H}_R(t) = \frac{1}{2} (B_0 - \omega) \hat{S}^z + \frac{1}{2} B_\perp \hat{S}^x - \frac{1}{2} B_\perp \left[\cos(-2\omega t) \hat{S}^x + \sin(-2\omega t) \hat{S}^y \right]. \quad (3.16)$$

The first two terms describe the Rabi oscillation discussed above. On the other hand, the last term oscillates fast with frequency 2ω , which is typically much larger than Ω_R . Such an oscillation is therefore too fast for the system to respond to, and its effect is negligible as the typical time scale of the system is fixed by the Rabi frequency Ω_R in (3.13). On this ground, assuming $\omega \approx B_0$, one often drops the fast oscillating term from the Hamiltonian (3.16) so that

$$\hat{H}(t) = B_\perp \sin(\omega t) \hat{S}^y + \frac{1}{2} B_0 \hat{S}^z \approx \frac{1}{2} B_\perp \left[\cos(\omega t) \hat{S}^x + \sin(\omega t) \hat{S}^y \right] + \frac{1}{2} B_0 \hat{S}^z. \quad (3.17)$$

Such an approximation is called the *rotating-wave approximation*. The approximation is valid as long as $B_0 \gg \Omega_R$.

3.2.2 Implementation of CNOT

CNOT gate is a vital gate operation in any quantum algorithms. Seemingly simple, it is not trivial to physically realize in a realistic system. A typical obstacle is that the Hamiltonian, in particular the coupling between two qubits, is restricted to certain limited forms. Here we take a few examples of the qubit-qubit interaction and see how one can use it to implement the CNOT gate. These examples give a general idea about what is required for the implementation of CNOT or similar gates in a given realistic physical system.

One of the most common form of the exchange interaction between two qubits is the so-called *Heisenberg exchange interaction*

$$\hat{H} = J(\hat{S}_1^x \hat{S}_2^x + \hat{S}_1^y \hat{S}_2^y + \hat{S}_1^z \hat{S}_2^z). \quad (3.18)$$

As one can see from its matrix representation

$$\hat{H} \doteq J \begin{bmatrix} 1 & & & \\ & -1 & 2 & \\ & 2 & -1 & \\ & & & 1 \end{bmatrix}, \quad (3.19)$$

it operates nontrivially only within the subspace spanned by $|01\rangle$ and $|10\rangle$ just like the SWAP gate discussed in Section 2.2.1—see Eq. (2.45). This is because the Heisenberg exchange coupling conserves the total angular momentum. More explicitly, up to a constant shift and multiplication factor, the Heisenberg exchange Hamiltonian equals to the SWAP gate

$$\frac{J + \hat{H}}{2J} \doteq \begin{bmatrix} 1 & & & \\ & 0 & 1 & \\ & 1 & 0 & \\ & & & 1 \end{bmatrix} \doteq \text{SWAP}. \quad (3.20)$$

The SWAP gate behaves like the NOT gate within the subspace spanned by $|01\rangle$ and $|10\rangle$, and hence the rotation around the x -axis—in the Bloch sphere corresponding to the subspace—by angle π results in the NOT gate. It implies that by tuning the exchange coupling constant and the operation time τ such that $J\tau = \pi/4$ achieves the SWAP gate (Loss & DiVincenzo, 1998)

$$\text{SWAP} = \exp \left[-\frac{i\pi}{4} (\hat{S}_1^x \hat{S}_2^x + \hat{S}_1^y \hat{S}_2^y + \hat{S}_1^z \hat{S}_2^z - 1) \right] \quad (3.21)$$

As pointed out in Section 2.2.1, the SWAP gate itself is not universal. The limitation originates from the fact that the Heisenberg exchange interaction conserves the total angular momentum. However, the $\sqrt{\text{SWAP}}$ gate is universal. One can construct the CZ gate by combining the $\sqrt{\text{SWAP}}$ gate with single-qubit gates as discussed in Section 2.2.1. It takes two Pauli X gates to construct the CNOT gate from the CZ gate as we recall the identity in Eq. (2.43).

Summing up, when the Heisenberg exchange interaction is available in a system of qubits, one can achieve the CZ gate using two $\sqrt{\text{SWAP}}$ gates and three single-qubit gates. If desired, one can apply two additional Hadamard gates to implement the CNOT gate.

Consider the Heisenberg exchange coupling.

```
In[1]:= op = MultiplyDot[S[1, All], S[2, All]]
Out[1]= S1X S2X + S1Y S2Y + S1Z S2Z
```

This is the matrix representation.

```
In[2]:= mat = Matrix[(1 + op) / 2];
mat // MatrixForm
Out[2]//MatrixForm=

$$\begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix}$$


In[3]:= opU = MultiplyExp[-I (Pi / 4) * (op - 1)]
opU // Elaborate
Out[3]= e-\frac{1}{4} i \pi (-1 + S_1^X S_2^X + S_1^Y S_2^Y + S_1^Z S_2^Z)

Out[4]= 
$$\frac{1}{2} + \frac{1}{2} S_1^X S_2^X + \frac{1}{2} S_1^Y S_2^Y + \frac{1}{2} S_1^Z S_2^Z$$


In[5]:= matU = Matrix[opU];
matU // MatrixForm
Out[5]//MatrixForm=

$$\begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix}$$

```

There are two additional types of qubit-qubit exchange interaction, the XY and Isign exchange interaction. The *XY exchange interaction*—also known as the *planar exchange interaction*—is described by the Hamiltonian

$$\hat{H} = J(\hat{S}_1^x \hat{S}_2^x + \hat{S}_1^y \hat{S}_2^y). \quad (3.22)$$

The matrix representation

$$\hat{H} \doteq \begin{bmatrix} 0 & & & \\ & 0 & 2 & \\ & 2 & 0 & \\ & & & 0 \end{bmatrix} \quad (3.23)$$

suggests that one can use the XY exchange interaction in an essentially the same way as the Heisenberg exchange interaction. An explicit construction is left for an exercise—see Problem 3.2.

Next, consider the XY exchange interaction.

```
In[1]:= op = MultiplyDot[S[1, {1, 2}], S[2, {1, 2}]]
Out[1]= S1X S2X + S1Y S2Y
```

```
In[=]:= mat = Matrix[op];
mat // MatrixForm
Out[=]//MatrixForm=

$$\begin{pmatrix} 0 & 0 & 0 & 0 \\ 0 & 0 & 2 & 0 \\ 0 & 2 & 0 & 0 \\ 0 & 0 & 0 & 0 \end{pmatrix}$$


```



```
In[=]:= opU = MultiplyExp[-I \phi op]
Out[=]= e^{-i \phi (S_1^x S_2^x + S_1^y S_2^y)}
```



```
In[=]:= Matrix[opU] // MatrixForm
Out[=]//MatrixForm=

$$\begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & \cos[2\phi] & -i \sin[2\phi] & 0 \\ 0 & -i \sin[2\phi] & \cos[2\phi] & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix}$$

```

The *Ising exchange interaction* has the Hamiltonian

$$\hat{H} = J \hat{S}_1^z \hat{S}_2^z, \quad (3.24)$$

where the z -component has no special meaning. Replacing the z -component with the x - or y -component has the same effect. The Ising exchange interaction allows for a more direct implementation of the CZ gate: Although a direct investigation is enough to see it, let us here take another view of the CZ gate for the heuristic purposes. Recall that the CZ gate is defined as

$$\text{CZ} = \sum_{x=0,1} |x\rangle \langle x| \otimes \hat{Z}^x = i \sum_x |x\rangle \langle x| \otimes e^{-i\pi x \hat{Z}/2} \quad (3.25)$$

As $x = 0, 1$ are the eigenvalues of $|1\rangle \langle 1| = (\hat{I} - \hat{Z})/2$,

$$\text{CZ} = e^{i\pi/4} \exp \left[-\frac{i\pi}{4} (\hat{Z} \otimes \hat{I} + \hat{I} \otimes \hat{Z} - \hat{Z} \otimes \hat{Z}) \right] \quad (3.26)$$

We note that the exponent involves the Ising exchange interaction between the two qubits. Therefore, one can implement the CZ gate with the first and second qubits as the control and target qubits, respectively, in terms of the Hamiltonian (up to an irrelevant global phase factor $e^{i\pi/4}$) of the form

$$\hat{H} = B(\hat{S}_1^z + \hat{S}_2^z) - J \hat{S}_1^z \hat{S}_2^z. \quad (3.27)$$

Note that the exchange coupling—apart from the single-qubit term—is of the Ising type. Putting $B = J$, the time-evolution operator governed by the Hamiltonian is given by

$$\hat{U}(t) = \exp \left[-i J t (\hat{S}_1^z + \hat{S}_2^z - \hat{S}_1^z \hat{S}_2^z) \right]. \quad (3.28)$$

Therefore, the CZ gate is achieved by tuning the parameters B and J and the operation time such that

$$J\tau = B\tau = \frac{\pi}{4}. \quad (3.29)$$

Let us now consider the Ising exchange interaction. Here we have introduced an additional single-qubit term controlling the second qubit.

```
In[1]:= op = S[1, 3] + S[2, 3] - S[1, 3] ** S[2, 3]
Out[1]= -S1z S2z + S1z + S2z

In[2]:= mat = Matrix[op];
mat // MatrixForm
Out[2]/MatrixForm=

$$\begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & -3 \end{pmatrix}$$


In[3]:= opU = MultiplyExp[-I φ op]
opU // Matrix // MatrixForm
Out[3]= e-i φ (-S1z S2z+S1z+S2z)

Out[4]/MatrixForm=

$$\begin{pmatrix} e^{-i\phi} & 0 & 0 & 0 \\ 0 & e^{-i\phi} & 0 & 0 \\ 0 & 0 & e^{-i\phi} & 0 \\ 0 & 0 & 0 & e^{3i\phi} \end{pmatrix}$$


In[5]:= opU = Exp[I Pi / 4] × MultiplyExp[-I Pi / 4 op]
opU // Matrix // MatrixForm
Out[5]= ei\pi/4 e-i\pi (-S1z S2z+S1z+S2z)

Out[6]/MatrixForm=

$$\begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & -1 \end{pmatrix}$$

```

3.3 Geometric/Topological Scheme

So far we have mostly assumed a piecewise-constant Hamiltonian (in the fixed or rotating frame). But the Hamiltonian may change continuously in time as well. When a system undergoes a cyclic adiabatic process starting from a particular eigenstate of the Hamiltonian, the system remains in the same energy level without making a transition to other energy levels. However, the quantum state of the system acquires a phase factor from two contributions. One is responsible for the usual dynamical accumulation and the other results from the geometric properties of the parameter space. The latter is called the *geometric phase* of the cyclic adiabatic process (Berry, 1984). When the energy level is degenerate and associated with a multi-dimensional eigen-subspace, the geometric phase becomes non-Abelian, that is, the quantum state evolves to another state within the subspace. The unitary transformation between the initial and final state is called the *non-Abelian geometric phase* (Wilczek & Zee, 1984). Non-Abelian geometric phase can be extended to any *cyclic evolution*, without restriction by the adiabatic condition (Aharonov & Anandan, 1987; Anandan, 1988).

The *geometric scheme* of quantum computation (or simply *geometric quantum computation* for short) is to implement the unitary gate operations by means of

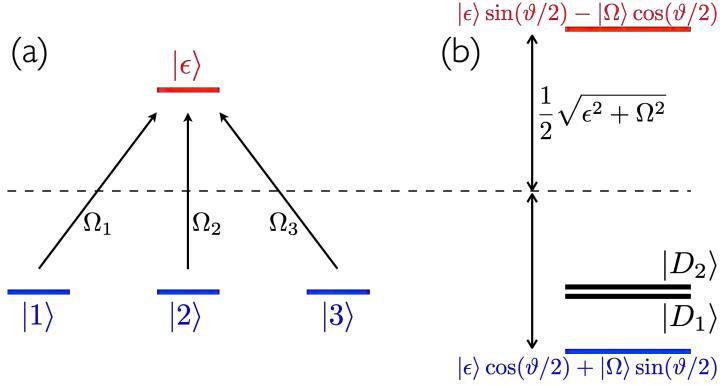


Figure 3.2: A schematic of the level structure of a toy model (see the text) for the geometric quantum computation. (a) The coupling between the ground-state levels and the excited-state level. (b) The eigenstates and corresponding eigenenergies of the model Hamiltonian. Here $|\Omega\rangle := \Omega^{-1} \sum_{j=1}^3 |j\rangle \Omega_j$, $\Omega := \sqrt{\Omega_1^2 + \Omega_2^2 + \Omega_3^2}$, and $\tan \vartheta := \Omega/\epsilon$. $|D_1\rangle$ and $|D_2\rangle$ denote the dark states.

the non-Abelian geometric phases (Zanardi & Rasetti, 1999; Sjöqvist *et al.*, 2012). The geometric scheme is stable against random fluctuations of the parameters as it depends on the geometric path in the parameter space rather than the detailed time-dependence of the parameters. In this section, we will give a brief overview of the geometric scheme.

3.3.1 A Toy Model

Let us start with a toy model (Choi, 2003). It consists of four levels $|1\rangle$, $|2\rangle$, $|3\rangle$ and $|\epsilon\rangle$. The level structure is depicted in Fig. 3.2 (a). The three ground states, $|1\rangle$, $|2\rangle$, $|3\rangle$, are all at energy zero. The excited state $|\epsilon\rangle$ has the (unperturbed) energy ϵ . Each ground state $|j\rangle$ is coupled with amplitudes Ω_j to the excited state $|\epsilon\rangle$, but the ground states are not directly coupled to each other. The model is governed by the Hamiltonian

$$\hat{H} = \epsilon |\epsilon\rangle \langle \epsilon| - \frac{1}{2} \sum_{j=1}^3 \Omega_j (|j\rangle \langle \epsilon| + |\epsilon\rangle \langle j|). \quad (3.30)$$

For simplicity, we have assumed that Ω_j are all real-valued. But the analysis below can be easily extended to the case with complex-valued Ω_j . For a system with non-degenerate ground-state levels, the model may arise in the interaction picture with suitable tuning of driving frequencies.

We will vary $\Omega_j(t)$ continuously in time, and the Hamiltonian $\hat{H}(t)$ will change accordingly through $\Omega_j(t)$. Before doing it, however, let us first examine the

instantaneous eigenstates of the Hamiltonian $\hat{H}(t)$ at a fixed instant t of time. We define a new state of superposition

$$|\Omega\rangle := \frac{1}{\Omega} \sum_{j=1}^3 |j\rangle \Omega_j, \quad (3.31)$$

where we have put $\Omega := \sqrt{\Omega_1^2 + \Omega_2^2 + \Omega_3^2}$. Then, the Hamiltonian reads as

$$\hat{H} = \epsilon |\epsilon\rangle \langle \epsilon| - \frac{1}{2} \Omega (|\Omega\rangle \langle \epsilon| + |\epsilon\rangle \langle \Omega|). \quad (3.32)$$

The Hamiltonian involves only two states $|\epsilon\rangle$ and $|\Omega\rangle$. Formally, it is a two-level Hamiltonian. The Hamiltonian is represented as

$$\hat{H} \doteq \begin{bmatrix} 0 & -\Omega/2 \\ -\Omega/2 & \epsilon \end{bmatrix} \quad (3.33)$$

in the basis $\{|\Omega\rangle, |\epsilon\rangle\}$. One can immediately identify two eigenstates

$$|\epsilon\rangle \cos(\vartheta/2) + |\Omega\rangle \sin(\vartheta/2), \quad |\epsilon\rangle \sin(\vartheta/2) - |\Omega\rangle \cos(\vartheta/2), \quad (3.34)$$

where $\tan \vartheta := \Omega/\epsilon$, with eigenenergies $(\epsilon \mp \sqrt{\epsilon^2 + \Omega^2})/2$. The eigenenergies of the Hamiltoinan are shown in Fig. 3.2 (b).

More interesting is that the two states $|\epsilon\rangle$ and $|\Omega\rangle$ —equivalently, the two eigenstates mentioned above—spans only part of the four-dimensional Hilbert space. There must be two more states. These additional states, which we denote by $|D_1\rangle$ and $|D_2\rangle$, do not appear in the Hamiltonian at all. They are completely decoupled from the rest. In this sense, $|D_1\rangle$ and $|D_2\rangle$ are called “dark states”. The state $|\Omega\rangle$, which couples to the excited-state $|\epsilon\rangle$, is called a “bright state”. Note that the dark states must be orthogonal to $|\Omega\rangle$ as well as $|\epsilon\rangle$. In other words, the set $\{|D_1\rangle, |D_2\rangle, |\Omega\rangle\}$ is another orthogonal basis for the space spanned by the ground states $|1\rangle, |2\rangle$, and $|3\rangle$. This relation between the dark and bright states is schematically illustrated in Fig. 3.3.

In this toy model, the two dark states, $|D_1\rangle$ and $|D_2\rangle$, span a degenerate subspace \mathcal{K} . It is of our primary interest as the energies of the dark states remain degenerate regardless of the values of Ω_j . It is convenient to take the parameterization

$$\Omega_1 = \Omega \sin \theta \cos \phi, \quad \Omega_2 = \Omega \sin \theta \sin \phi, \quad \Omega_3 = \Omega \cos \theta, \quad (3.35)$$

which leads to

$$|\Omega\rangle = |1\rangle \sin \theta \cos \phi + |2\rangle \sin \theta \sin \phi + |3\rangle \cos \theta. \quad (3.36)$$

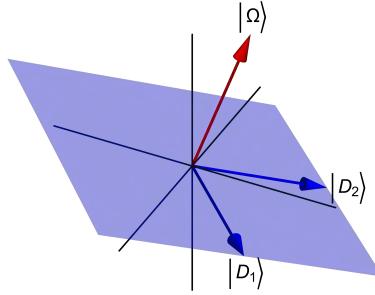


Figure 3.3: A schematic representation of the dark states of the toy model. The dark states $|D_1\rangle$ and $|D_2\rangle$ reside in the subspace orthogonal to the bright state $|\Omega\rangle$. The space spanned by the ground states $|1\rangle$, $|2\rangle$, and $|3\rangle$ are represented by the three-dimensional space \mathbb{R}^3 .

As already mentioned, for $|D_1\rangle$ and $|D_2\rangle$, one can choose any mutually orthogonal states from the subspace that is orthogonal to $|\Omega\rangle$. A convenient choice of the dark states is as follows

$$|D_1\rangle = |1\rangle \sin \phi - |2\rangle \cos \phi, \quad (3.37a)$$

$$|D_2\rangle = |1\rangle \cos \theta \cos \phi + |2\rangle \cos \theta \sin \phi - |3\rangle \sin \theta. \quad (3.37b)$$

It is important to note that the basis states $|D_1\rangle$ and $|D_2\rangle$ vary with parameters θ and ϕ (and hence with time t).

So far we have just constructed the time-dependent basis $|D_1(t)\rangle$ and $|D_2(t)\rangle$ —the time dependence of them is through $\theta(t)$ and $\phi(t)$. Suppose that $|\psi(t)\rangle$ is a *physical* state initially prepared in the dark-state subspace \mathcal{K} . We assume that the parameters change sufficiently slow so that the adiabatic condition is satisfied.^{3.1} Then $|\psi(t)\rangle$ always remains within \mathcal{K} . So we can expand $|\psi(t)\rangle$ in $|D_1(t)\rangle$ and $|D_2(t)\rangle$ at the same instant t ,

$$|\psi(t)\rangle = |D_1(t)\rangle c_1(t) + |D_2(t)\rangle c_2(t), \quad (3.38)$$

where $c_1(t)$ and $c_2(t)$ are complex coefficients. $|\psi(t)\rangle$ should satisfy the Schrödinger equation

$$i \frac{d}{dt} |\psi(t)\rangle = \hat{H}(t) |\psi(t)\rangle = 0. \quad (3.39)$$

^{3.1}We can remove this assumption by modulating the phases $\varphi_j(t)$ of the complex amplitudes $\Omega_j(t) = \Delta_j e^{i\varphi_j(t)}$. In this case, the adiabatic condition is not necessary.

The second equality follows from the fact that $\hat{H}(t)$ vanishes within \mathcal{K} , i.e., $\hat{H}(t)|D_j(t)\rangle = 0$. Putting (3.38) into (3.39), we get

$$\sum_j \left[|\dot{D}_j(t)\rangle c_j(t) + |D_j(t)\rangle \dot{c}_j(t) \right] = 0. \quad (3.40)$$

The over-dot indicates the derivative with respect to time t . Multiplying the above equation by $\langle D_i(t)|$ from the left, we obtain the differential equation for $c_j(t)$

$$\frac{d}{dt} \begin{bmatrix} c_1(t) \\ c_2(t) \end{bmatrix} + \begin{bmatrix} \langle D_1(t)|\dot{D}_1(t)\rangle & \langle D_1(t)|\dot{D}_2(t)\rangle \\ \langle D_2(t)|\dot{D}_1(t)\rangle & \langle D_2(t)|\dot{D}_2(t)\rangle \end{bmatrix} \begin{bmatrix} c_1(t) \\ c_2(t) \end{bmatrix} = 0. \quad (3.41)$$

Let us denote the 2×2 matrix in (3.41) by $\mathbf{A}(t)$ with elements

$$A_{ij}(t) := \langle D_i(t)| \frac{d}{dt} |D_j(t)\rangle. \quad (3.42)$$

Then, equation (3.41) reads as

$$\frac{d\mathbf{c}}{dt} + \mathbf{A}(t) \cdot \mathbf{c}(t) = 0, \quad (3.43)$$

where the vector \mathbf{c} denotes the coefficients c_1 and c_2 collectively, $\mathbf{c} := (c_1, c_2)^T$.

Let us further simplify the problem by considering a special case where we only vary ϕ but keep θ constant. Then we use the chain rule to rewrite the matrix $A(t)$ into the form

$$\mathbf{A}(t) = \frac{d\phi}{dt} \mathbf{A}^\phi \quad (3.44)$$

with the matrix \mathbf{A}^ϕ defined by the elements

$$A_{ij}^\phi := \langle D_i(\phi)| \frac{d}{d\phi} |D_j(\phi)\rangle. \quad (3.45)$$

Similarly, applying the chain rule to $d\mathbf{c}/dt$, we can rewrite Eq. (3.43) as

$$\frac{d\mathbf{c}}{d\phi} + \mathbf{A}^\phi(\phi) \cdot \mathbf{c}(\phi) = 0. \quad (3.46)$$

We have factored $d\phi/dt$ out of the equation. Now the equation has no explicit dependence on time t . The solution $\mathbf{c}(\phi)$ is completely determined by the geometric path in the parameter space.

Putting the expressions (3.37) into (3.45), we find that

$$\mathbf{A}^\phi = \begin{bmatrix} 0 & -\cos \theta \\ \cos \theta & 0 \end{bmatrix} = -i \cos \theta \mathbf{Y}, \quad (3.47)$$

where \mathbf{Y} is the Pauli Y matrix. After one cycle of variation from $\phi = 0$ to 2π , the coefficients become

$$\mathbf{c}(2\pi) = \exp(-2\pi \mathbf{A}^\phi) \mathbf{c}(0) = \mathbf{U}_y(-4\pi \cos \theta) \mathbf{c}(0), \quad (3.48)$$

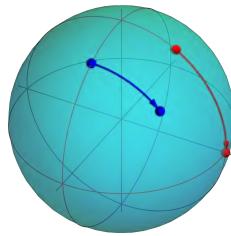


Figure 3.4: The rotation around the y -axis on the Bloch sphere based on the non-Abelian geometric phase.

where $\mathbf{U}_y(\varphi) = \exp(-i\mathbf{Y}\varphi/2)$ is the 2×2 matrix describing the rotation by angle φ around the y -axis in the Bloch space associated with $|D_1\rangle$ and $|D_2\rangle$. It is illustrated in Fig. 3.4. Through the modulation of the parameter ϕ and keeping the other parameter θ , we have achieved a particular type of quantum gates corresponding to the rotations around y -axis. By modulating the parameters along different paths in the parameter space, one can implement various single-qubit gates. One can show that such modulations in this toy model can implement rotations around two non-parallel axes can be implemented. Two-qubit gate operations can also be implemented in a similar way (Choi, 2003). These facts assert that universal quantum computation is possible based on the geometric scheme.

Consider a four-level system.

Let[Qudit, A, Dimension → 4]

Here is the standard basis of the Hilbert space associated with the system. We will regard

$|0\rangle$ as the excited level and $|1\rangle, |2\rangle, |3\rangle$ as the ground-state levels.

```
In[5]:= bs = Basis[A];
bs // LogicalForm
Out[5]= { |0_A>, |1_A>, |2_A>, |3_A> }
```

The ground states are coupled (each $|j\rangle$ with an amplitude Ω_j) to the excited level.

```
In[6]:= Let[Real, Ω, ε]
H = ε A[0 → 0] - (1/2) * PlusDagger@Sum[Ω[j] * A[0 → j], {j, 1, 3}]
Out[6]= ε (|0>⟨0|) + 1/2 (-(|1>⟨0|) Ω1 - (|0>⟨1|) Ω1 -
(|2>⟨0|) Ω2 - (|0>⟨2|) Ω2 - (|3>⟨0|) Ω3 - (|0>⟨3|) Ω3)
```

We choose a parameterization. This parameterization is convenient to normalize various states.

```
Let[Real, θ, φ]
Ω[1] = Sin[θ] × Cos[φ];
Ω[2] = Sin[θ] × Sin[φ];
Ω[3] = Cos[θ];
```

Here is the “bright” state in the parameterization.

```
In[5]:= ketΩ = Sum[Ket[A → j] * Ω[j], {j, 1, 3}]
Out[5]= Cos[θ] |3A⟩ + Cos[φ] |1A⟩ Sin[θ] + |2A⟩ Sin[θ] × Sin[φ]
```

Here is a choice of two “dark” states.

```
In[6]:= ketD1 = Ket[A → 1] * Sin[φ] - Ket[A → 2] * Cos[φ]
ketD2 =
Ket[A → 1] * Cos[θ] * Cos[φ] + Ket[A → 2] * Cos[θ] * Sin[φ] - Ket[A → 3] * Sin[θ]
Out[6]= -Cos[φ] |2A⟩ + |1A⟩ Sin[φ]
Out[7]= Cos[θ] × Cos[φ] |1A⟩ - |3A⟩ Sin[θ] + Cos[θ] |2A⟩ Sin[φ]
```

Let us check if they are indeed “dark” with respect to the Hamiltonian.

```
In[8]:= H ** ketD1
H ** ketD2
Out[8]= 0
Out[9]= 0
```

The two dark state must be orthogonal to the bright state. Let us check it.

```
In[10]:= new = {ketD1, ketD2, ketΩ};
Outer[Multiply, Dagger[new], new] // Simplify // MatrixForm
Out[10]//MatrixForm=

$$\begin{pmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{pmatrix}$$

```

Finally, we calculate the non-Abelian gauge potential.

```
In[11]:= dark = {ketD1, ketD2};
matA = Outer[Multiply, Dagger[dark], D[dark, φ]] // Simplify;
matA // MatrixForm
Out[11]//MatrixForm=

$$\begin{pmatrix} 0 & -\text{Cos}[θ] \\ \text{Cos}[θ] & 0 \end{pmatrix}$$

```

3.3.2 Geometric Phase

Let us now discuss the general case. Consider a cyclic process where the Hamiltonian changes in time through the control parameters $λ_μ$ with $μ = 1, 2, \dots$

$$\hat{H}(t) = \hat{H}(\boldsymbol{\lambda}(t)), \quad (3.49)$$

where we adopted a row-vector notation $\boldsymbol{\lambda} = (λ_1, \dots, λ_μ, \dots)$ to collectively denote the control parameters. When the Hamiltonian $\hat{H}(t)$ is the idealistic one in (3.1) for qubits, of course, the control parameters $\boldsymbol{\lambda}$ refer to the parameters $B_j^μ$ and $J_{ij}^μ$. However, the Hamiltonian for geometric quantum computation is usually much more general and does not contain elements manifesting a peculiar form of qubits. The logical qubits of a geometric scheme are often implicitly encoded into the subspace undergoing the cyclic evolution.

Let the states $|\alpha_j(t)\rangle$ ($j = 1, 2, \dots$) form an *instantaneous basis* of the Hilbert space \mathcal{H} . These states are often *chosen* to be the instantaneous eigenstates of the Hamiltonian $\hat{H}(t)$,

$$\hat{H}(t) |\alpha_j(t)\rangle = E_j(t) |\alpha_j(t)\rangle, \quad (3.50)$$

with time-dependent eigenenergies $E_j(t)$. But the choice is completely arbitrary as long as they are *cyclic*, $|\alpha_j(\tau)\rangle = |\alpha_j(0)\rangle$, and *smooth* enough to be differentiable with respect to time (and hence with respect to the parameters λ_μ). As the basis varies with time, the bases at different instants t and t' must be related to each other

$$|\alpha_j(t')\rangle = \sum_{ij} |\alpha_i(t)\rangle U_{ij}(t, t') \quad (3.51)$$

by a unitary matrix $U_{ij}(t, t') := \langle \alpha_i(t) | \alpha_j(t') \rangle$; see Eqs. (A.4) and (A.5). Choosing different bases at different times is analogous to choosing different reference frames at different positions as illustrated in Fig. 3.5. It is important to perceive the relation between different reference frames. Naturally, the unitary matrix $U(t)$ plays an important role when describing the dynamics in the time-dependent problem.

As the choice is arbitrary, the instantaneous basis states $|\alpha_j(t)\rangle$ are not directly related to the physical state. A physical state should satisfy the Schrödinger equation

$$i \frac{d}{dt} |\psi(t)\rangle = \hat{H}(t) |\psi(t)\rangle. \quad (3.52)$$

Suppose that a physical state $|\psi(t')\rangle$ at a given time t' is expanded in the instantaneous basis $|\alpha_j(t')\rangle$ as

$$|\psi(t')\rangle = \sum_j |\alpha_j(t')\rangle \langle \alpha_j(t') | \psi(t') \rangle. \quad (3.53)$$

At later time $t > t'$, it must be expanded in another instantaneous basis $|\alpha_i(t)\rangle$ in the form

$$|\psi(t)\rangle = \sum_i |\alpha_i(t)\rangle U_{ij}(t, t') \langle \alpha_j(t') | \psi(t') \rangle, \quad (3.54)$$

where $U_{ij}(t, t')$ is the unitary matrix in (3.51) that describes the basis change to $|\alpha_i(t)\rangle$ from $|\alpha_j(t')\rangle$. Putting (3.54) into the time-dependent Schrödinger equation (3.52), one can see that the unitary matrix $U(t, t')$ satisfies the dynamical equation

$$i \frac{\partial}{\partial t} U(t, t') = [H(t) - iA(t)] U(t, t'), \quad (3.55)$$

where $H(t)$ is the matrix representation of the Hamiltonian $\hat{H}(t)$ in the instantaneous basis $|\alpha_j(t)\rangle$, $H_{ij}(t) := \langle \alpha_i(t) | \hat{H}(t) | \alpha_j(t) \rangle$, and the matrix $A(t)$ has been defined by

$$A_{ij}(t) := \langle \alpha_i(t) | \frac{d}{dt} |\alpha_j(t)\rangle. \quad (3.56)$$

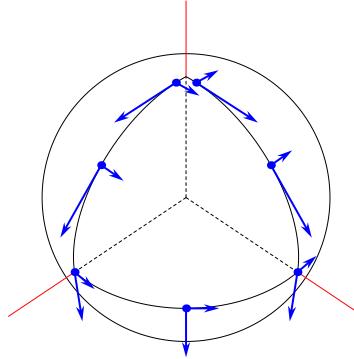


Figure 3.5: Illustration of a parallel transport on the unit sphere.

The additional term $-iA(t)$ in (3.55) is a non-inertial effect similar to the one we observed—see Eq. (3.10)—in the rotating frame in the Rabi oscillation. The solution to the equation (3.55) is formally given by

$$U(t, t') = \text{Texp} \left[- \int_{t'}^t ds \{ A(s) + iH(s) \} \right], \quad (3.57)$$

where T denotes the time ordering.^{3.2}

So far, everything has been completely general. For a geometric quantum computation, it is convenient to identify a dynamical subspace $\mathcal{K}(t) \subset \mathcal{H}$ such that

- (a) $\mathcal{K}(t)$ undergoes a *cyclic evolution*, $\mathcal{K}(\tau) = \mathcal{K}(0)$;
- (b) $\hat{H}(t)$ vanishes within $\mathcal{K}(t)$ at all time $0 < t < \tau$.

With these two conditions satisfied, the unitary matrix governing the evolution of the state vector $|\psi(t)\rangle$ in (3.54) is determined solely by the matrix $A(t)$,

$$U(t, t') = \text{Texp} \left[- \int_{t'}^t ds A(s) \right], \quad (3.58)$$

Furthermore, using the chain rule for the total derivative with respect to t , we can rewrite the matrix $A(t)$ into the form

$$A_{ij}(t) = \sum_{\mu} \frac{d\lambda_{\mu}}{dt} \langle \alpha_i | \frac{\partial}{\partial \lambda_{\mu}} | \alpha_j \rangle = \sum_{\mu} \frac{d\lambda_{\mu}}{dt} A_{ij}^{\mu}, \quad (3.59)$$

^{3.2}For a given matrix $M(t)$, the time-ordered exponential of $M(t)$ is defined by the series

$$\text{Texp} \left[\int_0^t ds M(s) \right] := I + \int_0^t ds M(s) + \int_0^t ds_1 \int_0^{s_1} ds_2 M(s_1)M(s_2) + \dots$$

Notice the upper limits of the integrals.

where we have put

$$A_{ij}^\mu := \langle \alpha_i | \frac{\partial}{\partial \lambda_\mu} | \alpha_j \rangle . \quad (3.60)$$

Putting $A(t)$ in (3.59) back to (3.58), one can see that the unitary matrix $U(\tau, 0)$ for the whole cycle is completely determined by the closed path in the parameter space that is traversed by the parameters $\lambda_\mu(t)$. That is, for a given loop \mathcal{C} in the parameter space, the unitary matrix $U(\tau, 0) = U(\mathcal{C})$ is given by

$$U(\mathcal{C}) = P \exp \left[- \sum_\mu \oint_{\mathcal{C}} d\lambda_\mu A^\mu \right] , \quad (3.61)$$

where P denotes path ordering. Putting it back into the evolution of the physical state in (3.54), we see that even without dynamical effect ($H = 0$ within the subspace \mathcal{K}) and after the Hamiltonian returns to the original value, $U(\mathcal{C})$ can make a non-trivial transitions of the states in the subspace \mathcal{K} .

To further understand \hat{A}^μ and $U(\mathcal{C})$, suppose that one chooses a different basis, say,

$$|\beta_j\rangle := \hat{V} |\alpha_j\rangle = \sum_i |\alpha_i\rangle V_{ij} , \quad (3.62)$$

where \hat{V} is a unitary operator and V is its matrix representation in the original basis $\{|\alpha_j\rangle\}$. Under the change of basis, the matrix A^μ transforms as

$$A^\mu \mapsto V^\dagger A^\mu V + V^\dagger \frac{\partial V}{\partial \lambda_\mu} , \quad (3.63)$$

the typical way a vector potential transforms under the gauge transformation in quantum mechanics. In this sense, A^μ is called the *non-Abelian gauge potential*, and describes the connection between the bases, $|\alpha_i(\lambda_\mu)\rangle$ and $|\alpha_j(\lambda_\mu + \delta\lambda_\mu)\rangle$, at infinitesimally different points along the path \mathcal{C} in the parameter space. The gauge connection A^μ and the corresponding non-Abelian geometric phase $U(\mathcal{C})$ are in close analogy with the parallel transport in curved space illustrated in Fig. 3.5.

In short, the geometric quantum computation is to implement the quantum gate operations by means of the unitary transformation $U(\mathcal{C})$ in (3.61). The computational space is identified by the subspace \mathcal{K} undergoing a cyclic evolution. Different choices of the closed path \mathcal{C} results in different quantum logic gates.

The *topological scheme* of quantum computation is a peculiar case of the geometric scheme. We will take a glimpse of an example of topological quantum computation in Section 6.5.

3.4 Measurement-Based Scheme

According to the fundamental postulates of quantum mechanics discussed at the very beginning of the book, the quantum state of a system changes as a consequence

of two different causes. One is the time-evolution governed by the Schrödinger equation. Both the dynamical and geometric schemes of quantum computation are inherently based on such time evolution. The other cause for the change of quantum states is the collapse of quantum states after measurement as dictated by Postulate 3. Can we also use it for quantum computation? At first glance, it may look absurd when considering the uncontrollable random nature of the collapse of a quantum state upon measurement. Nevertheless, it has recently been found that quantum computation is possible just by employing measurements in different bases provided the system is prepared in a special quantum state called the *cluster state* or, more generally, *graph state* (Raussendorf & Briegel, 2001; Raussendorf *et al.*, 2002). Such a scheme of quantum computation is called the *measurement-based quantum computation* or *one-way quantum computation*—“one-way” for the reason to be clear below. There are several methods to technically realize the measurement-based quantum computation. Here we introduce the elementary ideas and refer interested readers to the aforementioned articles and follow-up literature.

To get an idea how the measurement-based scheme works, first consider the quantum circuit model of the simple form

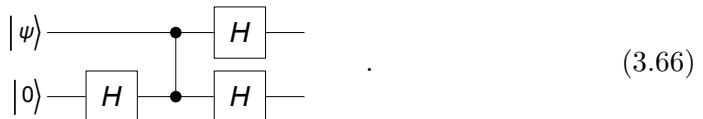


where the input state $|\psi\rangle = |0\rangle c_0 + |1\rangle c_1$ on the first qubit is arbitrary. Recall from Eq. (2.33) that the CNOT gate “copies” the logical basis states of the control qubit to the target qubit. It transforms the input state $|\psi\rangle \otimes |0\rangle$ to $|00\rangle c_0 + |11\rangle c_1$. The additional Hadamard gate on the first qubit leads to the final state

$$\frac{|0\rangle \otimes |\psi\rangle + |1\rangle \otimes (\hat{Z}|\psi\rangle)}{\sqrt{2}}. \quad (3.65)$$

When the first qubit is measured, the state of the second qubit is set to different states depending on the measurement outcome: When the outcome is 0, the state is identical to the input state of the first qubit. It is $\hat{Z}|\psi\rangle$ if the outcome is 1. In any case, the state coming out on the second qubit can be set to the (unknown) input state of the first qubit—with an additional operation \hat{Z} if necessary—because we know the measurement outcome.

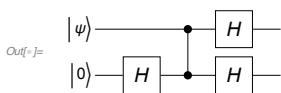
We can rewrite the quantum circuit model in (3.64) using relation (2.43) between the CNOT and CZ gates into the form



At this stage, there is no reason why we should prefer the CZ gate to the CNOT gate. However, we will see below that the CZ gate is suited better to the measurement-based scheme. More importantly, the Hadarmard gate followed by a measurement in the standard basis on the first qubit can be regarded as a measurement in the Pauli X basis. It is indeed achieved physically by rotating the measurement setup as we have discussed in Section 2.5. In effect, the measurement on the Pauli X basis on the first qubit has transfers a quantum state to the second qubit.

Consider the following quantum circuit model.

```
In[1]:= qc = QuantumCircuit[ProductState[S[1] → c@{0, 1}, "Label" → "|ψ⟩"],  
LogicalForm[Ket[], S[2]], S[2, 6], CZ[S[1], S[2]], S[{1, 2}, 6]]
```



This is the output state.

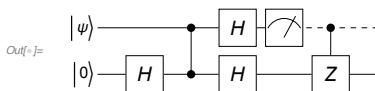
```
In[2]:= out = Elaborate[qc]  
Out[2]=  $\frac{c_0 |-\rangle}{\sqrt{2}} + \frac{c_0 |1_{S_1}\rangle}{\sqrt{2}} - \frac{c_1 |1_{S_1}1_{S_2}\rangle}{\sqrt{2}} + \frac{c_1 |1_{S_2}\rangle}{\sqrt{2}}$ 
```

As you can see here, when the first qubit is measured and the outcome is 0, then the second qubit is identical to the input state of the first qubit. In case the measurement outcome is 1, the second qubit is set to the initial state of the first qubit with the Pauli Z operated.

```
In[3]:= QuissoFactor[out, S[1]] // LogicalForm  
Out[3]=  $|0_{S_1}\rangle \otimes \left( \frac{c_0 |0_{S_2}\rangle}{\sqrt{2}} + \frac{c_1 |1_{S_2}\rangle}{\sqrt{2}} \right) + |1_{S_1}\rangle \otimes \left( \frac{c_0 |0_{S_2}\rangle}{\sqrt{2}} - \frac{c_1 |1_{S_2}\rangle}{\sqrt{2}} \right)$ 
```

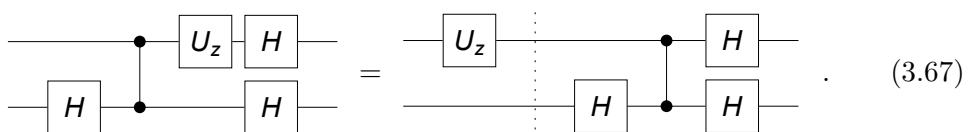
According to the above observation, we undo the byproduct operator depending on the outcome of the measurement on the first qubit.

```
In[4]:= qc2 = QuantumCircuit[qc, Measurement[S[1]], ControlledU[S[1], S[2, 3]]]
```



```
In[5]:= new = Elaborate[qc2] /. {c[0] * Conjugate[c[0]] + c[1] * Conjugate[c[1]] → 1};  
QuissoFactor[new, S[1]] // LogicalForm  
Out[5]=  $|1_{S_1}\rangle \otimes (c_0 |0_{S_2}\rangle + c_1 |1_{S_2}\rangle)$ 
```

Next, let us consider a slightly different quantum circuit model



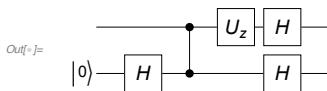
Compared with the previous quantum circuit model, it has a single-qubit rotation \hat{U}_z around the z -axis on the first qubit. In (3.67), the equality holds because \hat{U}_z commutes with the CZ gate. Applying the previous result to the quantum circuit model on the right-hand side of (3.67) immediately gives the output state

$$\frac{|0\rangle \otimes (\hat{U}_z |\psi\rangle) + |1\rangle \otimes (\hat{Z}\hat{U}_z |\psi\rangle)}{\sqrt{2}}. \quad (3.68)$$

Upon the measurement of the first qubit, the output state of the second qubit becomes either $\hat{U}_z |\psi\rangle$ or $\hat{Z}\hat{U}_z |\psi\rangle$ depending on the measurement outcome. In either case, the state $\hat{U}_z |\psi\rangle$ is achieved on the second qubit with an additional operation \hat{Z} if necessary. Again, the key point here is that the unitary operation \hat{U}_z followed by a measurement in the standard basis on the first qubit can be regarded as a measurement in the rotated basis—see Section 2.5. In the end, we have achieved the state with the unitary gate \hat{U}_z applied on it by means of (rotated) measurement.

Consider the following quantum circuit model.

```
In[1]:= Let[Real, φ]
qc = QuantumCircuit[LogicalForm[Ket[], S[2]], S[2, 6], CZ[S[1], S[2]], Rotation[φ, S[1, 3]], S[{1, 2}, 6]]
```



This shows how the logical basis states in the first qubit is transformed through the above quantum circuit model.

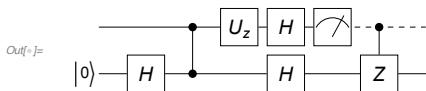
```
In[2]:= bs = Basis[S[1]];
out = qc ** bs // TrigToExp // Garner;
Thread[LogicalForm[bs] \[Rightarrow]
LogicalForm[QuissoFactor[#, S[1]] & /@ out, S@{1, 2}]] // TableForm
```

$$\begin{aligned} |\theta_{S_1}\rangle &\rightarrow |\theta_{S_1}\rangle \otimes \left(\frac{e^{-\frac{i\phi}{2}} |\theta_{S_2}\rangle}{\sqrt{2}} \right) + |\mathbf{1}_{S_1}\rangle \otimes \left(\frac{e^{\frac{i\phi}{2}} |\theta_{S_2}\rangle}{\sqrt{2}} \right) \\ |\mathbf{1}_{S_1}\rangle &\rightarrow |\theta_{S_1}\rangle \otimes \left(\frac{e^{\frac{i\phi}{2}} |\mathbf{1}_{S_2}\rangle}{\sqrt{2}} \right) + |\mathbf{1}_{S_1}\rangle \otimes \left(-\frac{e^{\frac{i\phi}{2}} |\mathbf{1}_{S_2}\rangle}{\sqrt{2}} \right) \end{aligned}$$

When the first qubit is measured, the output state of the second qubit depends on the measurement outcome. In either case, however, the input state of the first qubit with the unitary gate operated can be recovered in the second qubit with an operation of the Pauli Z if the measurement outcome is 1.

In accordance with the above argument, the byproduct operator can be fixed as follows.

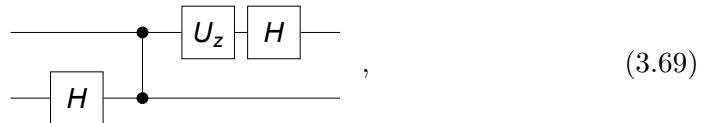
```
In[3]:= qc2 = QuantumCircuit[qc, Measurement[S[1]], ControlledU[S[1], S[2, 3]]]
```



```
In[~]:= new = qc2 ** bs // TrigToExp;
Thread[LogicalForm[bs] →
LogicalForm[QuissoFactor[#, S[1]] & /@ new, S@{1, 2}]] // TableForm
Out[~]/TableForm=
```

$$\begin{aligned} |\theta_{S_1}\rangle &\rightarrow |1_{S_1}\rangle \otimes \left(e^{-\frac{i\phi}{2}} |\theta_{S_2}\rangle \right) \\ |1_{S_1}\rangle &\rightarrow |\theta_{S_1}\rangle \otimes \left(e^{\frac{i\phi}{2}} |1_{S_2}\rangle \right) \end{aligned}$$

As the inverse of the Hadamard gate equals to itself, $\hat{H}^{-1} = \hat{H}$, one can remove the second Hadamard gate on the second qubit by applying \hat{H} . It leads to the quantum circuit model



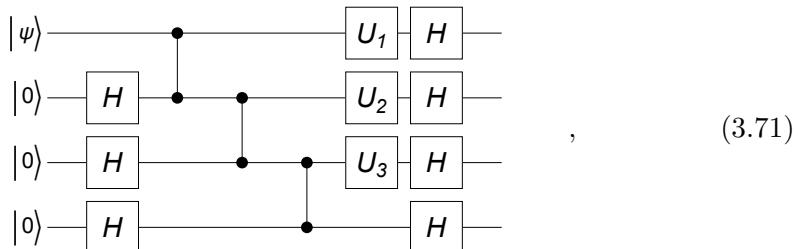
Accounting for the extra Hamamard gate that we applied, we get the output state

$$\frac{1}{\sqrt{2}} \sum_{x_1=0,1} |x_1\rangle \otimes (\hat{H} \hat{Z}^{x_1} \hat{U}_z |\psi\rangle) \quad (3.70)$$

for the above quantum circuit model. Now the quantum circuit model requires no quantum gates once an entanglement is created by the CZ gate. Only measurement is needed. The cost is that the Hadamard gate appears on the output qubit as a byproduct. However, we will see below that the trailing Hadamard gate on the second qubit plays an important role to achieve an arbitrary single-qubit rotation in the measurement-based scheme.

3.4.1 Single-Qubit Rotations

Le us now inspect how to implement arbitrary single-qubit rotations solely based on measurement. Consider the following quantum circuit model



where the label U_j denotes the single-qubit rotation $\hat{U}_z(\phi_j)$ around the z -axis by angle ϕ_j . The output state from the above quantum circuit model can be analyzed by consecutively applying the result in (3.70) from (3.69). It reads as

$$\sum_{x_1,x_2,x_3} |x_1\rangle \otimes |x_2\rangle \otimes |x_3\rangle \otimes \left(\hat{Z}^{x_3} \hat{U}_z(\phi_3) \hat{H} \hat{Z}^{x_2} \hat{U}_z(\phi_2) \hat{H} \hat{Z}^{x_1} \hat{U}_z(\phi_1) |\psi\rangle \right). \quad (3.72)$$

Using the identity $\hat{H}\hat{Z}\hat{H} = \hat{X}$, the final state on the forth qubit can be written as

$$|\Psi\rangle_4 = \hat{Z}^{x_3}\hat{U}_z(\phi_3)\hat{X}^{x_2}\hat{U}_x(\phi_2)\hat{Z}^{x_1}\hat{U}_z(\phi_3)|\psi\rangle. \quad (3.73)$$

Further, using the identities $\hat{X}\hat{Z}\hat{X} = -\hat{Z}$ and $\hat{Z}\hat{X}\hat{Z} = -\hat{X}$, we arrive at the expression

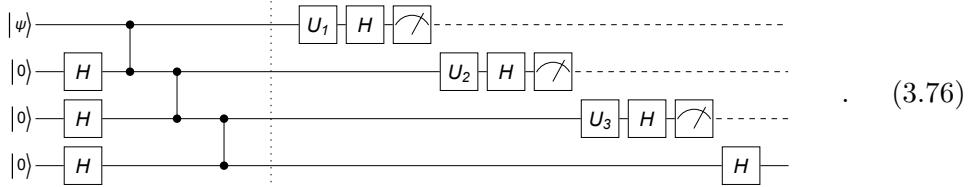
$$|\Psi\rangle_4 = \hat{Z}^{x_3}\hat{X}^{x_2}\hat{Z}^{x_1}\hat{U}_z((-1)^{x_2}\phi_3)\hat{U}_x((-1)^{x_1}\phi_2)\hat{U}_z(\phi_1)|\psi\rangle. \quad (3.74)$$

The key point to notice here is that upon the measurement on the first three qubits, the forth qubit takes on the state with a single-qubit rotation

$$\hat{U}_z((-1)^{x_2}\phi_3)\hat{U}_x((-1)^{x_1}\phi_2)\hat{U}_z(\phi_1) \quad (3.75)$$

operated on the initial state $|\psi\rangle$ of the first qubit. The rotation consists of three rotations around two perpendicular axes. They are sufficient to realize any arbitrary single-qubit rotation—see Section 2.1.3.

The rotation in (3.75) is not deterministic since the rotation direction—clockwise or anticlockwise—depends on the outcomes x_1, x_2, x_3 of the measurement on the first three qubits. We can make it deterministic by delaying the operations on the second after the measurement on the first qubit and the operation on the third qubit after the measurement on the second qubit. Consider a modified quantum circuit model as follows



Suppose that we want to implement a single-qubit Euler rotation^{3.3}

$$\hat{U}(\alpha, \beta, \gamma) := \hat{U}_z(\alpha)\hat{U}_x(\beta)\hat{U}_z(\gamma). \quad (3.77)$$

We first set $\phi_1 = \gamma$. Then, depending the measurement outcome x_1 on the first qubit, we set $\phi_2 = (-1)^m\beta$. Similarly, depending on the measurement outcome x_2 on the second qubit, we set $\phi_3 = (-1)^{x_2}\alpha$. The final state on the forth qubit will become

$$|\Psi\rangle_4 = \hat{Z}^{x_3}\hat{X}^{x_2}\hat{Z}^{x_1}\hat{U}(\alpha, \beta, \gamma)|\psi\rangle \quad (3.78)$$

with the single-qubit rotation fixed *deterministically*. There are still undesired operations, $\hat{Z}^{x_3}\hat{X}^{x_2}\hat{Z}^{x_1}$. However, these operations are irrelevant at the end of quantum computation, and do not have to be corrected. For example, if the final state is measured in the logical basis, \hat{Z} does not affect the readout at all, and the effect of \hat{X} can be handled by a classical post-processing.

^{3.3}Here we have adopted a different convention for the Euler rotation. One of the most common convention in physics is to use the combination, $\hat{U}_z(\alpha)\hat{U}_y(\beta)\hat{U}_z(\gamma)$.

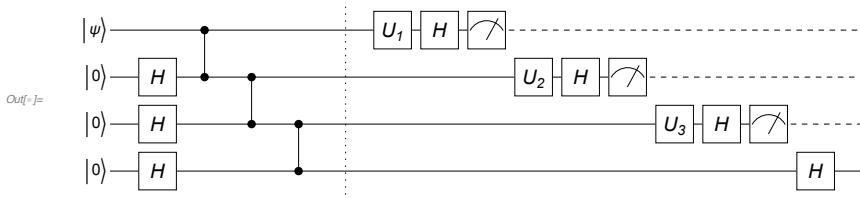
Let us construct a quantum circuit model for an arbitrary single-qubit rotation in the measurement-based scheme.

This will simplify the analysis below.

```
Let[Real, c, φ]
```

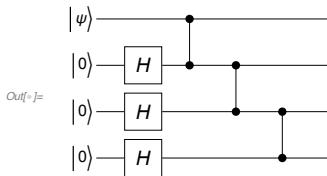
Here is the overall quantum circuit model. It is built in two steps divided by the vertical dashed line. The first step corresponds to the preparation, and the second to the operation.

```
In[7]:= qc = QuantumCircuit[ProductState[S[1] → {c[0], c[1]}, "Label" → Ket["ψ"]],  
LogicalForm[Ket[], S@{2, 3, 4}], S[{2, 3, 4}, 6],  
CZ[S[1], S[2]], CZ[S[2], S[3]], CZ[S[3], S[4]], "Separator",  
Rotation[φ[1], S[1, 3], "Label" → "U1"], S[1, 6], Measurement[S[1]],  
Rotation[φ[2], S[2, 3], "Label" → "U2"], S[2, 6], Measurement[S[2]],  
Rotation[φ[3], S[3, 3], "Label" → "U3"], S[3, 6], Measurement[S[3]], S[{4}, 6]]
```



For the sake of analysis, we build the first part of the above quantum circuit model separately.

```
In[8]:= qc0 = QuantumCircuit[  
ProductState[S[1] → {c[0], c[1]}, "Label" → Ket["ψ"]],  
LogicalForm[Ket[], S@{2, 3, 4}],  
S[{2, 3, 4}, 6], CZ[S[1], S[2]], CZ[S[2], S[3]], CZ[S[3], S[4]]]
```



Perform a measurement on the first qubit in a properly rotated basis.

```
In[9]:= out1 = QuantumCircuit[qc0, Rotation[φ[1], S[1, 3], "Label" → "U1"],  
S[1, 6], Measurement[S[1]]] // Elaborate;  
x1 = Readout[out1, S[1]]
```

```
Out[9]= 1
```

Depending on the outcome x_1 out of the measurement on the first qubit, rotate suitable the measurement setup for the second qubit and perform the measurement.

```
In[10]:= out2 = QuantumCircuit[out1, Rotation[(-1) ^ x1 * φ[2], S[2, 3], "Label" → "U2"],  
S[2, 6], Measurement[S[2]]] // Elaborate;  
x2 = Readout[out2, S[2]]
```

```
Out[10]= 0
```

Similarly, depending on the measurement outcome x_2 on the second qubit, adjust the measurement setup for the third qubit and take the measurement.

```
In[=]:= out3 = QuantumCircuit[out2, Rotation[(-1) ^ x2 * φ[3], S[3, 3], "Label" → "U3"], S[3, 6], Measurement[S[3]], S[4, 6]] // Elaborate;
out3 = out3 /. {c[0] ^ 2 + c[1] ^ 2 → 1};
x3 = Readout[out3, S[3]];

Out[=]= 0
```

Check the result.

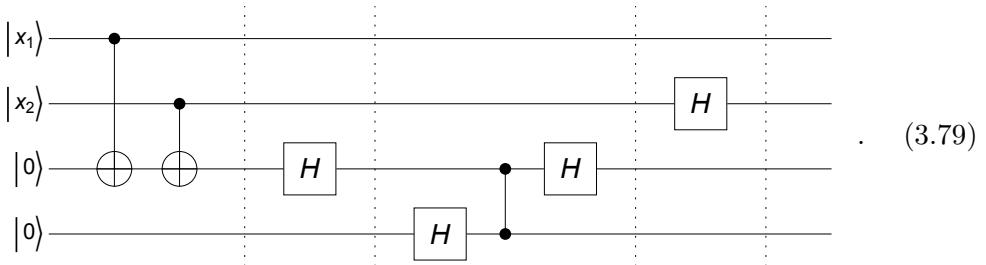
```
In[=]:= expr = MultiplyPower[S[4, 3], x3] **
MultiplyPower[S[4, 1], x2] ** MultiplyPower[S[4, 3], x1] **
Rotation[φ[3], S[4, 3]] ** Rotation[φ[2], S[4, 1]] **
Rotation[φ[1], S[4, 3]] ** (Ket[S[1] → x1, S[2] → x2, S[3] → x3] **
ProductState[S[4] → {c[0], c[1]}]) // Elaborate

Out[=]= |1S1⟩ (Cos[φ1/2] (c0 Cos[φ2/2] - i c1 Sin[φ2/2]) + Sin[φ1/2] (-i c0 Cos[φ2/2] + c1 Sin[φ2/2])) ×
(Cos[φ3/2] - i Sin[φ3/2]) + |1S11S4⟩
(Cos[φ1/2] (-c0 Cos[φ2/2] + i c0 Sin[φ2/2]) + Sin[φ1/2] (-i c1 Cos[φ2/2] + c1 Sin[φ2/2])) ×
(Cos[φ3/2] + i Sin[φ3/2])
```

```
In[=]:= out3 - expr
Out[=]= 0
```

3.4.2 CNOT Gate

Let us now construct a quantum circuit model to implement a CNOT gate in the measurement-based scheme. There are several ways. Here we adopt an example described in the quantum circuit model



The first two CNOT gates bring the input state—recall Eq. (2.55) and in (3.24)—to

$$|x_1\rangle \otimes |x_2\rangle \otimes |x_1 \oplus x_2\rangle \otimes |0\rangle \quad (3.80)$$

at the instant indicated by the first vertical dotted line. At the second vertical dotted line, the state becomes

$$|x_1\rangle \otimes |x_2\rangle \otimes (\hat{H} |x_1 \oplus x_2\rangle) \otimes |0\rangle. \quad (3.81)$$

Now we use (3.69), in effect, to transfer the state stored in the third qubit to the forth qubit. It gives the state

$$\begin{aligned} \sum_{y_3} |x_1\rangle \otimes |x_2\rangle \otimes |y_3\rangle \otimes (\hat{H}\hat{Z}^{y_3}\hat{H}|x_1 \oplus x_2\rangle) \\ = \sum_{y_3} |x_1\rangle \otimes |x_2\rangle \otimes |y_3\rangle \otimes (\hat{X}^{y_3}|x_1 \oplus x_2\rangle). \end{aligned} \quad (3.82)$$

Finally, applying the Hadamard gate on the second qubit—see Eq. (2.19)—leads to

$$\sum_{y_2, y_3} |x_1\rangle \otimes |y_2\rangle \otimes |y_3\rangle \otimes (\hat{X}^{y_3}|x_1 \oplus x_2\rangle)(-1)^{y_2 x_2}. \quad (3.83)$$

The phase factor -1 occurs when $x_2 = y_2 = 1$. By inspection, one can see that the same factor arises when one applies \hat{Z} on both the first and forth qubits. This leads to the final state out of the quantum circuit model as follows

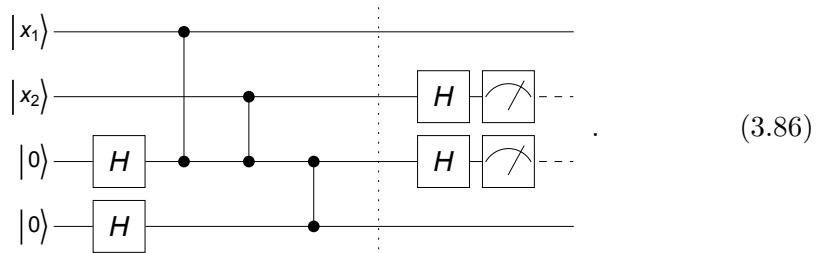
$$|\Psi\rangle_{\text{out}} = \sum_{y_2, y_3} (\hat{Z}^{y_2}|x_1\rangle) \otimes |y_2\rangle \otimes |y_3\rangle \otimes (\hat{X}^{y_3}\hat{Z}^{y_2}|x_1 \oplus x_2\rangle). \quad (3.84)$$

When one performs measurement on the second and third qubits, depending on the outcomes y_2 and y_3 , the final state stored in the first and forth qubits is given by

$$|\Psi\rangle_{14} = (\hat{Z}^{y_2}|x_1\rangle) \otimes (\hat{X}^{y_3}\hat{Z}^{y_2}|x_1 \oplus x_2\rangle). \quad (3.85)$$

Let us ignore the byproduct operators, \hat{X} and/or \hat{Z} , for the moment. Then, we see that in this quantum circuit model, the first qubit plays a role of the control qubit. The input state of the target qubit is put in the second qubit, and the output state is supposed to appear in the forth qubit.

Using the relation (2.43) between the CZ and CNOT gates, we can rewrite the quantum circuit model in (3.79) into the form

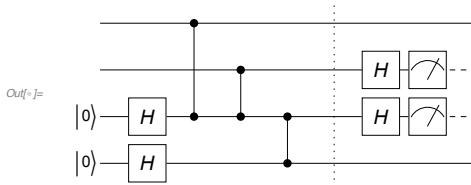


It is clear that once the entanglement among the four qubits, then it is sufficient to perform measurement on the second and third qubits to implement the CNOT gate. One has to apply \hat{Z} on both the first and forth qubit when $y_2 = 1$ and \hat{X} as well if $y_3 = 1$. But again, in many cases, the additional step is not necessary or can be done with a simple classical post-processing.

Here we demonstrate the CNOT gate based on the measurement-based scheme.

Let us consider the following quantum circuit model. The first qubit plays a role of the control qubit. The input state of the target qubit enters the second qubit and comes out on the forth qubit.

```
In[5]:= qc1 = QuantumCircuit[LogicalForm[Ket[], S@{3, 4}],
  S[{3, 4}, 6], CZ[S[1], S[3]], CZ[S[2], S[3]], CZ[S[3], S[4]],
  "Separator", S[{2, 3}, 6], Measurement[S@{2, 3}]]
```



This shows how the logical basis states on the first two qubits are transformed to the first and forth qubit. The result is not exactly what we expect from the CNOT gate since we have not corrected the byproduct operators.

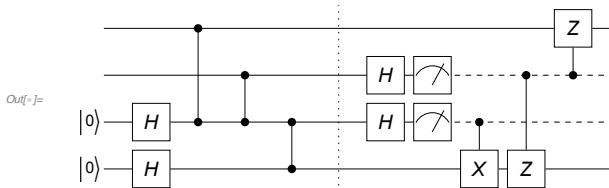
```
In[6]:= bs = Basis[S@{1, 2}];
out = qc1 ** bs;
Thread[LogicalForm[bs] → LogicalForm[out, S@{1, 2, 3, 4}]] // TableForm
```

Out[6]//TableForm=

$ 0_{S_1} 0_{S_2}\rangle \rightarrow 0_{S_1} 1_{S_2} 0_{S_3} 0_{S_4}\rangle$
$ 0_{S_1} 1_{S_2}\rangle \rightarrow 0_{S_1} 0_{S_2} 1_{S_3} 0_{S_4}\rangle$
$ 1_{S_1} 0_{S_2}\rangle \rightarrow 1_{S_1} 0_{S_2} 0_{S_3} 1_{S_4}\rangle$
$ 1_{S_1} 1_{S_2}\rangle \rightarrow 1_{S_1} 0_{S_2} 1_{S_3} 1_{S_4}\rangle$

The following quantum circuit model includes the correction of the byproduct operators.

```
In[7]:= qc2 = QuantumCircuit[qc1, ControlledU[S[3], S[4, 1]],
  ControlledU[S[2], S[4, 3]], ControlledU[S[2], S[1, 3]]]
```



We see that the output state stored on the first and forth qubit is the state we expect.

```
In[8]:= bs = Basis[S@{1, 2}];
out = qc2 ** bs;
Thread[LogicalForm[bs] → LogicalForm[out, S@{1, 2, 3, 4}]] // TableForm
```

Out[8]//TableForm=

$ 0_{S_1} 0_{S_2}\rangle \rightarrow 0_{S_1} 1_{S_2} 1_{S_3} 0_{S_4}\rangle$
$ 0_{S_1} 1_{S_2}\rangle \rightarrow 0_{S_1} 1_{S_2} 1_{S_3} 1_{S_4}\rangle$
$ 1_{S_1} 0_{S_2}\rangle \rightarrow 1_{S_1} 1_{S_2} 1_{S_3} 1_{S_4}\rangle$
$ 1_{S_1} 1_{S_2}\rangle \rightarrow 1_{S_1} 1_{S_2} 0_{S_3} 0_{S_4}\rangle$

In the above example, both the input and output states of the control qubit are stored in the same qubit whereas the target state is transferred from one to another qubit. In many practical cases, especially in the middle of quantum computation, it is more convenient to transfer the target qubit to another qubit as well. It can be achieved by using similar methods with 10 or 15 qubits (Raussendorf & Briegel, 2001; Raussendorf *et al.*, 2003).

3.4.3 Graph States

For the measurement-based quantum computation, the crucial resource is the state prepared by the quantum circuit elements on the left of the vertical dashed line in (3.76). Once the state is prepared, from then on, quantum logic gates can be implemented solely by measurements in rotated bases. Such a state is called a *graph state*. More specifically, consider a graph \mathcal{G} where a set of vertices are connected with edges. At each vertex is located a qubit. Let \hat{U}_{ab} denotes the CZ gate on the two qubits located at the vertex a and b . To generate the graph state $|\mathcal{G}\rangle$ associated with the graph \mathcal{G} , first prepare each qubit in the state $|+\rangle := (|0\rangle + |1\rangle)/\sqrt{2}$. Then for each pair $\langle ab \rangle$ of vertices a and b connected by an edge, apply the CZ gate \hat{U}_{ab} on the corresponding qubits. In short,

$$|\mathcal{G}\rangle := \prod_{\langle ab \rangle} \hat{U}_{ab} |+\rangle^{\otimes n}, \quad (3.87)$$

where the product is over all edges in the graph.

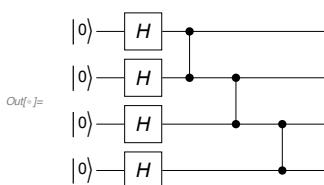
A graph state is created by applying the CZ gate on the two qubits linked by the graph.

For example, consider a linear graph.

```
In[7]:= gr = Graph[{S[1] <-> S[2], S[2] <-> S[3], S[3] <-> S[4]}, VertexLabels -> "Index"]
Out[7]=
```

The corresponding graph state is created by the following quantum circuit model.

```
In[8]:= qc = QuantumCircuit[LogicalForm[Ket[], S@{1, 2, 3, 4}], gr]
```



```
In[5]:= vec = Elaborate[qc];
vec // LogicalForm
Out[5]= 
$$\frac{1}{4} \left| 0_{S_1} 0_{S_2} 0_{S_3} 0_{S_4} \right\rangle + \frac{1}{4} \left| 0_{S_1} 0_{S_2} 0_{S_3} 1_{S_4} \right\rangle + \frac{1}{4} \left| 0_{S_1} 0_{S_2} 1_{S_3} 0_{S_4} \right\rangle - \frac{1}{4} \left| 0_{S_1} 0_{S_2} 1_{S_3} 1_{S_4} \right\rangle +$$


$$\frac{1}{4} \left| 0_{S_1} 1_{S_2} 0_{S_3} 0_{S_4} \right\rangle + \frac{1}{4} \left| 0_{S_1} 1_{S_2} 0_{S_3} 1_{S_4} \right\rangle - \frac{1}{4} \left| 0_{S_1} 1_{S_2} 1_{S_3} 0_{S_4} \right\rangle + \frac{1}{4} \left| 0_{S_1} 1_{S_2} 1_{S_3} 1_{S_4} \right\rangle +$$


$$\frac{1}{4} \left| 1_{S_1} 0_{S_2} 0_{S_3} 0_{S_4} \right\rangle + \frac{1}{4} \left| 1_{S_1} 0_{S_2} 0_{S_3} 1_{S_4} \right\rangle + \frac{1}{4} \left| 1_{S_1} 0_{S_2} 1_{S_3} 0_{S_4} \right\rangle - \frac{1}{4} \left| 1_{S_1} 0_{S_2} 1_{S_3} 1_{S_4} \right\rangle -$$


$$\frac{1}{4} \left| 1_{S_1} 1_{S_2} 0_{S_3} 0_{S_4} \right\rangle - \frac{1}{4} \left| 1_{S_1} 1_{S_2} 0_{S_3} 1_{S_4} \right\rangle + \frac{1}{4} \left| 1_{S_1} 1_{S_2} 1_{S_3} 0_{S_4} \right\rangle - \frac{1}{4} \left| 1_{S_1} 1_{S_2} 1_{S_3} 1_{S_4} \right\rangle$$

```

The same state can be directly obtained using `GraphState`.

```
In[6]:= new = GraphState[gr]
Out[6]= 
$$\frac{|-\rangle}{4} - \frac{1}{4} \left| 1_{S_1} 1_{S_2} \right\rangle - \frac{1}{4} \left| 1_{S_1} 1_{S_2} 1_{S_3} 1_{S_4} \right\rangle + \frac{|1_{S_1}\rangle}{4} + \frac{1}{4} \left| 1_{S_1} 1_{S_2} 1_{S_3} \right\rangle -$$


$$\frac{1}{4} \left| 1_{S_1} 1_{S_2} 1_{S_4} \right\rangle + \frac{1}{4} \left| 1_{S_1} 1_{S_3} \right\rangle - \frac{1}{4} \left| 1_{S_1} 1_{S_3} 1_{S_4} \right\rangle + \frac{1}{4} \left| 1_{S_1} 1_{S_4} \right\rangle - \frac{1}{4} \left| 1_{S_2} 1_{S_3} \right\rangle +$$


$$\frac{|1_{S_2}\rangle}{4} + \frac{1}{4} \left| 1_{S_2} 1_{S_3} 1_{S_4} \right\rangle + \frac{1}{4} \left| 1_{S_2} 1_{S_4} \right\rangle - \frac{1}{4} \left| 1_{S_3} 1_{S_4} \right\rangle + \frac{|1_{S_3}\rangle}{4} + \frac{|1_{S_4}\rangle}{4}$$

```

```
In[7]:= vec - new // Simplify
```

```
Out[7]= 0
```

Graph states are not only useful as a valuable resource for measurement-based quantum computation but also interesting in their own right. Graph states exhibit a high degree of entanglement. They have been studied extensively for the structure of bi-partite and multi-partite entanglement. One can also construct a class of quantum error-correction codes (see Chapter 6) based on graph states. It allows for reliable storage and processing of quantum information in a fault-tolerant way. All these features of graph states can be attributed to the following property: For each vertex a in \mathcal{G} , define an operator

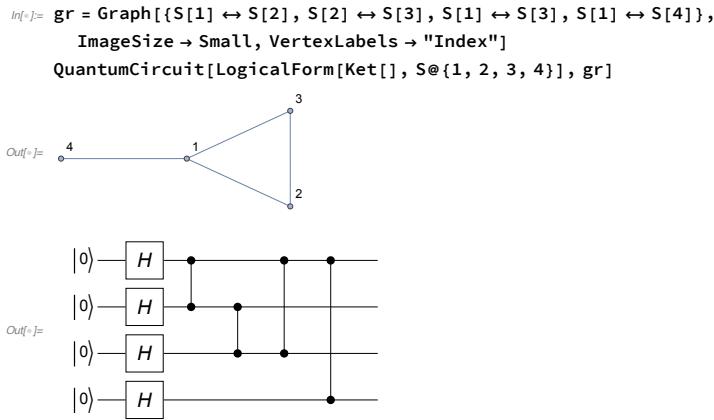
$$\hat{C}_a := \hat{X}_a \prod_{b \in \mathcal{N}_a} \hat{Z}_b, \quad (3.88)$$

where the product is over the set \mathcal{N}_a of all vertices b connected to a . \hat{C}_a is often called the *corelation operator* of the vertex a . The graph state $|\mathcal{G}\rangle$ associated with \mathcal{G} is the unique simultaneous eigenstate with eigenvalue +1 of every \hat{C}_a . In other words,

$$\hat{C}_a |\mathcal{G}\rangle = |\mathcal{G}\rangle \quad (3.89)$$

for all vertcies a in the graph \mathcal{G} . The operators \hat{C}_a are said to *stabilize* the graph state. The stabilizer formalism that we will discuss in Section 6.3 provides powerful tools to exploit the stabilizing operators.

As another example, consider the following graph and corresponding quantum circuit model.



This is the graph state associated with the above graph.

```
In[=]:= vec = GraphState[gr]  
Out[=]= 
$$\frac{1}{4} |\_ \_ \_ \_ \rangle - \frac{1}{4} |1_{S_1} 1_{S_2} \rangle - \frac{1}{4} |1_{S_1} 1_{S_2} 1_{S_3} \rangle + \frac{|1_{S_1}\rangle}{4} + \frac{1}{4} |1_{S_1} 1_{S_2} 1_{S_3} 1_{S_4} \rangle +$$
  

$$\frac{1}{4} |1_{S_1} 1_{S_2} 1_{S_4} \rangle - \frac{1}{4} |1_{S_1} 1_{S_3} \rangle + \frac{1}{4} |1_{S_1} 1_{S_3} 1_{S_4} \rangle - \frac{1}{4} |1_{S_1} 1_{S_4} \rangle - \frac{1}{4} |1_{S_2} 1_{S_3} \rangle -$$
  

$$\frac{1}{4} |1_{S_2} 1_{S_3} 1_{S_4} \rangle + \frac{|1_{S_2}\rangle}{4} + \frac{1}{4} |1_{S_2} 1_{S_4} \rangle + \frac{|1_{S_3}\rangle}{4} + \frac{1}{4} |1_{S_3} 1_{S_4} \rangle + \frac{|1_{S_4}\rangle}{4}$$

```

Here are the correlation operators associate with the vertices in the graph.

```
In[=]:= gnr = Stabilizer[gr]  
Out[=]= {Sx1 Sz2 Sz3 Sz4, Sz1 Sx2 Sz3, Sz1 Sz2 Sx3, Sz1 Sx4}
```

The graph state is the simultaneous eigenstate with eigenvalue +1 of all correlation operators.

```
In[=]:= (gnr ** vec) / vec  
Out[=]= {1, 1, 1, 1}
```

Mathematically, one can associate a graph state with any graph. However, such a state is difficult to generate in realistic systems. It requires applying the CZ gate on spatially separated qubits. *Cluster states* are a special subclass of graph states. The underlying graph is an d -dimensional square grid. The regular connectivity of the underlying graph makes the associated state far more feasible. Indeed, a number of physical methods to generate cluster states have been proposed.

3.5 Spin-Boson Model*

In many architectures of quantum computers, the coupling between qubits are indirectly achieved through a bosonic mode shared by the qubits. Common examples include the quantum computers based on superconducting qubits and trapped ions. The bosonic mode can also be used for a quantum non-demolition (QND) measurement of quantum states as in the quantum computers based on

superconducting circuits. Here we discuss the properties of such a spin-boson model at the elementary level.

Let us first examine the elementary properties of the model with a single-qubit coupled to a bosonic mode. The Hamiltonian is given by

$$\hat{H} = \omega \hat{a}^\dagger \hat{a} + \frac{1}{2} \Omega \hat{S}^z + g(\hat{a}^\dagger + \hat{a}) \hat{S}^x, \quad (3.90)$$

where \hat{a} and \hat{a}^\dagger are the annihilation and creation operators, respectively, of the bosonic mode.

Denote the qubits by S and the cavity mode by a.

```
Let[Qubit, S]
Let[Boson, a]
```

This is the Hamiltonian.

```
In[7]:= opH = Dagger[a] ** a + Ω S[3] / 2 + g (Dagger[a] + a) ** S[1]
Out[7]= a^\dagger a + g (a S^x + a^\dagger S^x) + Ω S^z / 2
```

Here is a *truncated* basis.

```
In[8]:= bs = Basis[{a, S}];
LogicalForm@bs
Out[8]= { |0_a 0_S⟩, |0_a 1_S⟩, |1_a 0_S⟩, |1_a 1_S⟩, |2_a 0_S⟩,
          |2_a 1_S⟩, |3_a 0_S⟩, |3_a 1_S⟩, |4_a 0_S⟩, |4_a 1_S⟩, |5_a 0_S⟩, |5_a 1_S⟩ }
```

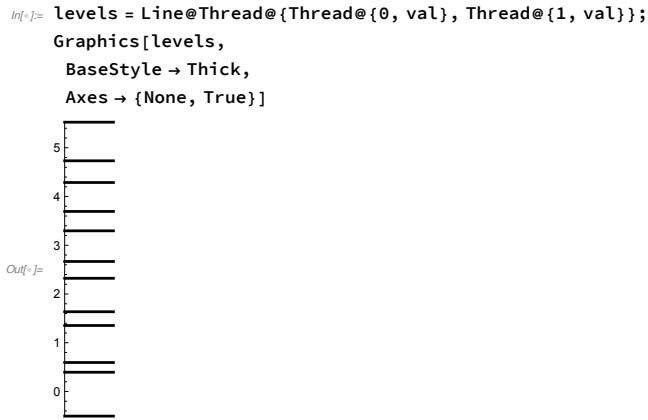
Here is the matrix representation of the Hamiltonian (showing only part of it), and the eigen-energies for a particular set of parameters. The matrix representation is not exact as the basis has been truncated.

```
In[9]:= Ω = 1;
g = 1 / 10;
matH = Matrix[opH];
matH[[;; 6, ;; 6]] // MatrixForm
val = N@Eigenvalues[matH]
Out[9]/MatrixForm=

$$\begin{pmatrix} \frac{1}{2} & 0 & 0 & \frac{1}{10} & 0 & 0 \\ 0 & -\frac{1}{2} & \frac{1}{10} & 0 & 0 & 0 \\ 0 & \frac{1}{10} & \frac{3}{2} & 0 & 0 & \frac{1}{5\sqrt{2}} \\ \frac{1}{10} & 0 & 0 & \frac{1}{2} & \frac{1}{5\sqrt{2}} & 0 \\ 0 & 0 & 0 & \frac{1}{5\sqrt{2}} & \frac{5}{2} & 0 \\ 0 & 0 & \frac{1}{5\sqrt{2}} & 0 & 0 & \frac{3}{2} \end{pmatrix}
Out[9]= \{ 5.52494, 4.7328, 4.2874, 3.69409, 3.29609, 2.66746,
          2.32239, 1.63601, 1.35389, 0.594847, -0.505013, 0.395102 \}$$

```

Here you can see the (approximate) energy levels of the spin-boson model.



...

Next we demonstrate the QND measurement of quantum states using the bosonic mode.

...

We demonstrate a basic method to induce a coupling between two qubits by sharing the single bosonic mode.

$$\hat{H} = \omega \hat{a}^\dagger \hat{a} + \frac{1}{2} \Omega \sum_{j=1}^2 \hat{S}_j^z + g(\hat{a}^\dagger + \hat{a}) \sum_j \hat{S}_j^x \quad (3.91)$$

...

Problems

- 3.1. As the single-parameter controlled Hamiltonian in Eq. (3.16) has two fictitious magnetic fields rotating in opposite senses, it seems to be a matter of choice to pick either of the two. Argue physically why it does not work to choose the frame rotating in the clockwise sense.
- 3.2. Consider a system of two qubits interacting with each other through the XY exchange coupling in (3.22). For the given coupling constant J , find the way to physically implement the SWAP gate by tuning the operation time τ . If necessary, you can apply additional single-qubits gates. Ignore a global phase factor (if any). Once the SWAP gate is carried out, you can combine the $\sqrt{\text{SWAP}}$ gate and single-qubit gates to construct the CZ gate—see Section 2.2.1.

- 3.3. Consider a toy model similar to the one described in Eq. (3.30) and Fig. 3.2, but with two ground-state levels:

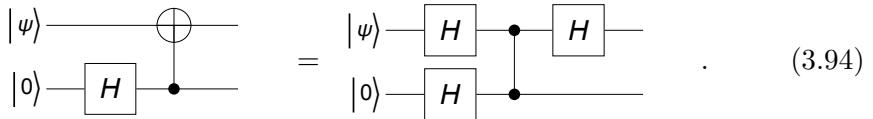
$$\hat{H} = \epsilon |\epsilon\rangle\langle\epsilon| - \frac{1}{2} \sum_{j=1}^2 (\Omega_j |j\rangle\langle j| + h.c.). \quad (3.92)$$

Assume that

$$\Omega_1 = \Omega \cos(\theta/2)e^{-i\phi/2}, \quad \Omega_2 = \Omega \sin(\theta/2)e^{+i\phi/2} \quad (\theta, \phi \in \mathbb{R}). \quad (3.93)$$

- (a) Find all eigenstates and corresponding eigenvalues of the Hamiltonian \hat{H} . Show that there exists an eigenstate $|D\rangle$ the eigenvalue of which is always zero regardless of Ω_1 and Ω_2 . $|D\rangle$ is the “dark state” of the model.
- (b) Calculate the (Abelian) gauge potential $A^\phi(\theta, \phi)$ for fixed θ for the one-dimensional subspace spanned by $|D\rangle$. Plot A^ϕ as a function of θ and ϕ .
- (c) Calculate the (Abelian) geometric phase $U(\mathcal{C})$ for the path \mathcal{C} such that θ is fixed and ϕ changes from 0 to 2π .

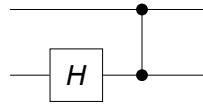
- 3.4. Consider the following quantum circuit model



Show that the output state is given by

$$\frac{|0\rangle \otimes |\psi\rangle + |1\rangle \otimes (\hat{X}|\psi\rangle)}{\sqrt{2}}. \quad (3.95)$$

- 3.5. Consider the following quantum circuit model:



- (a) Suppose that the input state is $(|0\rangle c_0 + |1\rangle c_1) \otimes |0\rangle$. Show that the output state is given by $|0\rangle \otimes |+\rangle c_0 + |1\rangle \otimes |-\rangle c_1$, where $|\pm\rangle := (|0\rangle \pm |1\rangle)/\sqrt{2}$ is the eigenstates (with eigenvalues ± 1) of the Pauli X operator.
- (b) Find the output state for the input state $(|0\rangle c_0 + |1\rangle c_1) \otimes |1\rangle$.

- 3.6. Let $\hat{U}_\mu(\phi)$ be the single-qubit rotation around the μ -axis by angle ϕ in the Pauli space. Show

(a) that

$$\hat{X}\hat{U}_z(\phi)\hat{X} = \hat{U}_z(-\phi), \quad (3.96)$$

where \hat{X} is the Pauli X operator; and

(b) that

$$\hat{H}\hat{U}_z(\phi)\hat{H} = \hat{U}_x(\phi), \quad (3.97)$$

where \hat{H} is the Hadamard operator.

Chapter 4

Quantum Algorithms

- August 9, 2021 (v1.17)

In Chapter 2, we have discussed how to carry out arbitrary computation by composing elementary quantum logic gates. In Chapter 3, we have seen what is required to physically implement those elementary quantum logic gates. The discussions are enough to establish physical and realistic models of quantum computation. However, so far, we have disregarded an important issue, that is, the efficiency of the implementations. Quantum computers turn out to be technically hard to build and error rates are still a fundamental concern for quantum computers while aforementioned calculations can be performed, in principle, on classical computers anyway. Why should quantum computation be attractive?

Not surprisingly, it was Peter Shor's quantum factorization algorithm ([Shor, 1994, 1997](#)) that brought quantum computation to such great attention even of the public at the turn of the millennium. The factorization of large numbers was the first practically important task that is not feasible on a classical computer but can be performed efficiently on a quantum computer. In this chapter, we explore several elementary examples of quantum algorithms that efficiently solve the problems that are known to be exponentially hard with classical algorithms. Some of them may be of little use for practical applications. Nevertheless, these examples are still interesting as one can take a glimpse of ideas and features behind quantum algorithms distinguished from classical algorithms through them. In the discussion, included is quantum teleportation. It is a quantum communication protocol rather than a quantum algorithm. Nonetheless, we include it here because it is a simple yet fascinating example demonstrating what one can do with quantum states that is not possible at all with classical information.

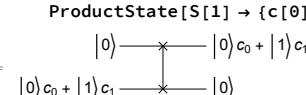
Throughout the chapter, we keep using the same notation in Chapter 2, Eq. (2.3) in particular.

4.1 Quantum Teleportation

Quantum teleportation is a communication protocol^{4.1} to send quantum information to a distant party utilizing a pre-shared pair of entangled particles. The protocol has attracted public interest as it resembles (fictional) teleportation. Teleportation in science fiction (hypothetically) transports matter or energy across space instantly. Quantum teleportation does not transfer physical objects but only quantum information. Quantum teleportation is also conceptually distinguished from telefax. Telefax makes a copy of the document and sends the duplicated copy. The original document remains at its origin. In quantum teleportation, a quantum state is transferred and disappears from the original system.

Of course, transmitting quantum information will be trivial if there is a reliable quantum channel between the parties. In quantum teleportation, it is assumed that there is no quantum channel available. The task will be straightforward as well if the two parties are in direct contact with each other. For example, one can simply use the SWAP gate in Eq. (2.44). Another example is to use a measurement-based method for transferring a quantum state in Eq. (3.64). In quantum teleportation, the two parties are at such a distance that no joint operation is attainable.

When one has access to both qubits, a SWAP gate for example is enough to send state from one qubit to the other.

```
In[5]:= Let[Complex, qc]
qc =
  QuantumCircuit[{LogicalForm[Ket[], S[1]], ProductState[S[2] \[Rule] {c[0], c[1]}]}, 
    SWAP[S[1], S[2]], {LogicalForm[Ket[], S[2]], 
      ProductState[S[1] \[Rule] {c[0], c[1]}]}, "PortSize" \[Rule] 2]
Out[5]= 
In[6]:= out = QuissoFactor@ExpressionFor[qc]
Out[6]= (c0 |0_{S_1}\rangle + c1 |1_{S_1}\rangle) \otimes |0_{S_2}\rangle
```

Quantum teleportation is one of the first quantum information protocols that have vividly illustrated the significance of quantum entanglement as a valuable resource. In this section, we discuss the physical principle behind quantum teleportation and demonstrate its protocol. Another closely related quantum communication protocol is the so-called *superdense coding*. As the idea and protocol are in parallel with the quantum teleportation protocol, here we do not discuss it and refer the readers to the original work ([Bennett & Wiesner, 1992](#)).

^{4.1}In this sense, quantum teleportation may not be classified rigorously as a quantum algorithm. Anyhow, it is included here because it is not only interesting in its own right but also used as part of various quantum algorithms.

4.1.1 Nonlocality in Entanglement

Quantum entanglement is a key resource in quantum teleportation as we will see shortly. However, it also reveals an intriguing nature of quantum mechanics, that is, *nonlocality*. Quantum entanglement and accompanying nonlocality is a largely unexpected consequence of the superposition principle of quantum states which was pointed out first by Einstein *et al.* (1935). Before we discuss an implementation of quantum teleportation, here we take a brief look at the nonlocal property buried in quantum entanglement.

Suppose that Alice and Bob share a pair of qubits that is in an entangled state

$$|\Psi\rangle = \frac{|0\rangle \otimes |0\rangle + |1\rangle \otimes |1\rangle}{\sqrt{2}} \quad (4.1)$$

Recall that it can be generated using the quantum entangler circuit in Section 2.2.1. When Bob measures his qubit, the measurement readout is just random and can be either 0 or 1. However, if Alice performs a measurement on her qubit anytime before Bob's measurement, Bob's result is completely fixed by the result of Alice's measurement.

This is the quantum entangler circuit. It generates a maximally entangled state.

```
In[7]:= qc = QuantumCircuit[LogicalForm[Ket[], S@{1, 2}], S[1, 6], CNOT[S[1], S[2]]]
Out[7]=
```

Here is the output state. It is indeed a maximally entangled state.

```
In[8]:= vec = ExpressionFor[qc];
vec // LogicalForm
Out[8]=
```

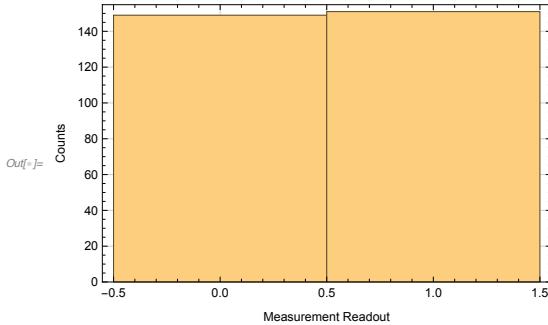
$$\frac{|0_{S_1}0_{S_2}\rangle}{\sqrt{2}} + \frac{|1_{S_1}1_{S_2}\rangle}{\sqrt{2}}$$

Now Bob measure on his qubit.

```
In[9]:= qc2 = QuantumCircuit[qc, "Spacer", Measurement@S[2]]
Out[9]=
```

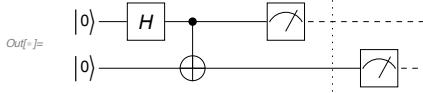
This shows that the measurement outcome is just random.

```
In[7]:= val = Table[out = Elaborate[qc2]; Readout[out, S[2]], {300}];
Histogram[val, FrameLabel -> {"Measurement Readout", "Counts"}]
```



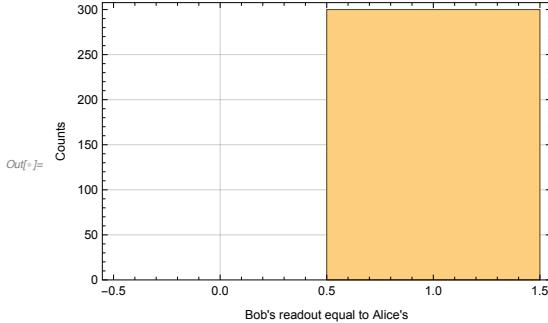
Here Alice, possessing the first qubit, measures her qubit before Bob measures his qubit.

```
In[8]:= qc3 = QuantumCircuit[qc, "Spacer",
Measurement@S[1], "Separator", Measurement@S[2]]
```



This shows that the measurement outcomes of Alice's and Bobs' are perfectly correlated. This illustrates that Bob's measurement results have affected by Alice measurement.

```
In[9]:= val = Boole@Table[out = Elaborate[qc2];
Equal @@ Readout[out, S[{1, 2}], {300}];
Histogram[val, {-0.5, 1.5, 1},
FrameLabel -> {"Bob's readout equal to Alice's", "Counts"}]
```



The above conclusion holds however far Alice and Bob are separated and however soon Bob measures after Alice does. Somehow Alice's measurement affects Bob's measurement “instantaneously”. It seemingly violates Einstein's special theory of relativity, which dictates that nothing can travel faster than light. This forced many people to hesitate to believe in quantum mechanics. That was until John Bell proposed an experimental test of an inequality that was later named after him (Bell, 1966). An experimental result violating Bell's inequality would mean that quantum mechanics is not a local realistic theory. Aspect *et al.* (1981) reported actual experiments that supported the nonlocality of quantum mechanics. Since Bell's pioneering work, many other inequalities have been proposed for

	(\hat{Z}, \hat{Z})	(\hat{Z}, \hat{X})	(\hat{X}, \hat{Z})	(\hat{X}, \hat{X})
$(+1, +1)$				
$(+1, -1)$		0		
$(-1, +1)$			0	
$(-1, -1)$	0			P

Table 4.1: Hardy’s test of nonlocality without inequality. The first row lists the combinations of observables to measure. The first column lists possible measurement outcomes. The three zeros indicate the known facts about the state of the given system: the measurement of the combination never gives the corresponding outcome. Then, a local realistic argument would predict that the probability P for measurement (\hat{X}, \hat{X}) to give $(-1, -1)$ is zero.

the tests of nonlocality of quantum mechanics. One of the best known is the Clauser-Horne-Shimony-Holt inequality ([Clauser *et al.*, 1969](#)).

Nonlocality tests has attracted renewed interest when [Greenberger *et al.* \(1990\)](#) proposed a new test *without inequality* using three spin-1/2 particles (i.e., three qubits). It is now known as the GHZ test after the authors. Unlike Bell-type nonlocality tests, it does not use inequality but is based on equality. Later, [Hardy \(1992, 1993\)](#) proposed a similar test wihtout inequality for two qubits. Hardy’s test is interesting as it works for almost all entangled states. The GHZ test uses a particular entangled state. Here we introduce Hardy’s test. We follow the arguments by [Goldstein \(1994\)](#) rather than Hardy’s original work.

Alice and Bob are spatially separated. Suppose that they share a pair of qubits in an entangled state of the form^{4.2}

$$|\Psi\rangle = \frac{|0\rangle \otimes |0\rangle + |0\rangle \otimes |1\rangle + |1\rangle \otimes |0\rangle}{\sqrt{3}}. \quad (4.2)$$

Each person chooses either \hat{Z} or \hat{X} and measures the selected observable on her/his qubit. (i) If both Alice and Bob choose \hat{Z} , then there is no chance for both to get outcome -1 because $|\Psi\rangle$ does not include the state $|1\rangle \otimes |1\rangle$. (ii) If Alice measures \hat{Z} and Bob measures \hat{X} , there occurs no event that Alice gets 1 and Bob gets -1 . This becomes clear when we rewrite $|\Psi\rangle$ in the form

$$|\Psi\rangle = \frac{\sqrt{2}}{\sqrt{3}} |0\rangle \otimes |+\rangle + \frac{1}{\sqrt{3}} |1\rangle \otimes |0\rangle, \quad (4.3)$$

^{4.2}Here we assume a particular entangled state for a simple introduction, but Hardy’s test has almost no restriction.

where $|\pm\rangle = (|0\rangle \pm |1\rangle)/\sqrt{2}$ are the eigenstates of \hat{X} belonging to the eigenvalues ± 1 . If Alice's qubit is in $|0\rangle$, then Bob's qubit is definitely in $|+\rangle$. They will never get the outcomes 1 and -1 , respectively. (iii) If Alice measures \hat{X} while Bob measures \hat{Z} , they will never get the respective outcomes -1 and 1 for a similar reason. This becomes clear by putting $|\psi\rangle$ in the form

$$|\Psi\rangle = \frac{1}{\sqrt{3}}|0\rangle \otimes |1\rangle + \frac{\sqrt{2}}{\sqrt{3}}|+\rangle \otimes |0\rangle. \quad (4.4)$$

The expected measurement results are summarized in Table 4.1.

Now, what if both Alice and Bob measure \hat{X} on their qubits? What would a *local realistic* argument predict given the known facts (i)–(iii)? Einstein *et al.* (1935) argues, “If, without in any way disturbing a system, we can predict with certainty the value of a physical quantity, then there exists an element of physical reality corresponding to this physical quantity.” This is called *reality* requirement. They also argue, “Since at the time of measurement the two systems no longer interact, no real change can take place in the second system in consequence of anything that may be done to the first system.” This is called *locality* requirement. Let us deduce the result of measurement combination (\hat{X}, \hat{X}) respecting the above two requirements. The fact (i) implies that either Alice's or Bob's qubit must be in the state $|0\rangle$. Suppose that Alice's qubit is in $|0\rangle$. Then, the fact (ii) implies that Bob's qubit must be in $|+\rangle$. In turn, this means that measuring (\hat{X}, \hat{X}) they will never observe the outcome $(-1, -1)$. Next consider the case where Bob's qubit is in $|0\rangle$. Then, the fact (iii) implies that Alice's qubit must be in $|+\rangle$. In turn, this means that measurement of (\hat{X}, \hat{X}) will never give $(-1, -1)$. Therefore, in any case, the probability for Alice and Bob to get the outcome $(-1, -1)$ from measurement of (\hat{X}, \hat{X}) is zero.

What does quantum mechanics predict? The probability for the outcome $(-1, -1)$ from measurement of (\hat{X}, \hat{X}) is given by

$$P = |\langle - - |\Psi\rangle|^2 = \frac{1}{12}. \quad (4.5)$$

It contradicts the above local realistic argument! It implies that quantum theory is nonlocal.

4.1.2 Implementation of Quantum Teleportation

Now let us turn back to the quantum teleportation protocol. Suppose that Bob wants to send one bit of quantum information, say, $|\psi\rangle_C = |0\rangle_C \psi_0 + |1\rangle_C \psi_1$ stored in Charlie's qubit, to Alice residing in a place far away from Bob (and Charlie). It is important to note that the quantum state $|\psi\rangle$ is unknown. The no-cloning theorem (see Section 1.3.1) prevents a naive approach such as copying and transmitting the quantum state.

To teleport a quantum state, Alice and Bob need to share an entangled pair of qubits in advance. Suppose that the shared pair of qubits is in the state

$$|\beta_0\rangle_{AB} = \frac{|00\rangle_{AB} + |11\rangle_{AB}}{\sqrt{2}}, \quad (4.6)$$

which is one of the four Bell states. Recall that the entanglement is generated by a combination of the Hadamard and CNOT gate—see Section 2.2.1. Thus the initial state $|\Psi\rangle$ of the three qubits at the beginning of the procedure is given by

$$|\Psi\rangle = |\beta_0\rangle_{AB} \otimes |\psi\rangle_C = \frac{|000\rangle\psi_0 + |110\rangle\psi_0 + |001\rangle\psi_1 + |111\rangle\psi_1}{\sqrt{2}} \quad (4.7)$$

Rewriting the parts consisting of the qubits B and C in the Bell basis—the basis consisting of the four Bell states—leads to

$$\begin{aligned} |\Psi\rangle = & (|0\rangle\psi_0 + |1\rangle\psi_1)_A \otimes |\beta_0\rangle_{BC} + (|1\rangle\psi_0 + |1\rangle\psi_0)_A \otimes |\beta_1\rangle_{BC} \\ & - (|1\rangle\psi_0 - |1\rangle\psi_0)_A \otimes |\beta_2\rangle_{BC} + (|0\rangle\psi_0 - |1\rangle\psi_1)_A \otimes |\beta_3\rangle_{BC} \end{aligned} \quad (4.8)$$

The crucial point here is that the state of A in the first term is identical to the quantum state $|\psi\rangle$, and those in the rest are also closely related to $|\psi\rangle$ by the Pauli operators. More explicitly, one can rewrite the total state vector as

$$\begin{aligned} |\Psi\rangle = & |\psi\rangle_A \otimes |\beta_0\rangle_{BC} + (\hat{S}_A^x |\psi\rangle_A) \otimes |\beta_1\rangle_{BC} \\ & + (i\hat{S}_A^y |\psi\rangle_A) \otimes |\beta_2\rangle_{BC} + (\hat{S}_A^z |\psi\rangle_A) \otimes |\beta_3\rangle_{BC} \end{aligned} \quad (4.9)$$

Now Bob performs a measurement on the two qubits B and C , owned by Bob, in the *Bell basis*. The measurement will yield outcomes $\mu = 0, 1, 2, 3$ and collapse the total state vector into the corresponding term, $\hat{S}_A^\mu |\psi\rangle_A \otimes |\beta_\mu\rangle$ (there is an additional factor of i for $\mu = 2$ but the global phase factor is physically irrelevant). The remaining task is for Bob to inform Alice of the outcome μ so that Alice recover the desired state $|\psi\rangle$ by operating the inverse operator \hat{S}_A^μ on her qubit. The information about the measurement outcome amounts to two bits, and requires only a classical channel for transmission. In short, the quantum teleportation protocol consists of the following steps:

1. Alice and Bob generate an entangled pair of qubits, A and B , and share the pair between them. This can be done anytime before the procedure actually starts. Bob prepares a quantum state to send in a separate qubit C .
2. Bob makes a Bell measurement, i.e., the measurement in the Bell basis, on his two qubits B and C .
3. Bob sends the two-bit information of the measurement outcome to Alice through a classical communication channel.

4. Alice operates a proper inverse operator to recover the desired quantum state.

Here is a simulation of the quantum teleportation protocol using Q3.

The initial state of the total system at the beginning of the protocol.

```
In[1]:= Let[Complex, ψ]
vec = Ket[] × ψ[0] + Ket[S[3] → 1] × ψ[1];
vec // LogicalForm
Out[1]= |0s3⟩ ψ0 + |1s3⟩ ψ1

In[2]:= tot = BellState[S@{0, 2}, 0] ** vec;
tot // LogicalForm
Out[2]= 
$$\frac{|0s_0 0s_2 0s_3\rangle \psi_0}{\sqrt{2}} + \frac{|1s_0 1s_2 0s_3\rangle \psi_0}{\sqrt{2}} + \frac{|0s_0 0s_2 1s_3\rangle \psi_1}{\sqrt{2}} + \frac{|1s_0 1s_2 1s_3\rangle \psi_1}{\sqrt{2}}$$

```

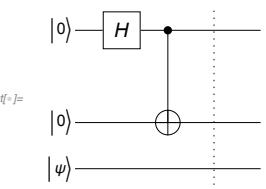
The total state is rewritten in the Bell basis for qubits S[2, None] and S[3, None].

```
In[3]:= bs = BellState[S@{2, 3}];
prj = Dyad[#, #] & /@ bs;
QuissoFactor /@ (prj ** tot) // TableForm
Out[3]/TableForm=

$$\begin{aligned}
&\frac{(|0s_2 0s_3\rangle + |1s_2 1s_3\rangle) \otimes (|0s_0\rangle \psi_0 + |1s_0\rangle \psi_1)}{2\sqrt{2}} \\
&\frac{(|0s_2 1s_3\rangle + |1s_2 0s_3\rangle) \otimes (|1s_0\rangle \psi_0 + |0s_0\rangle \psi_1)}{2\sqrt{2}} \\
&\frac{(-|0s_2 1s_3\rangle + |1s_2 0s_3\rangle) \otimes (|1s_0\rangle \psi_0 - |0s_0\rangle \psi_1)}{2\sqrt{2}} \\
&\frac{(|0s_2 0s_3\rangle - |1s_2 1s_3\rangle) \otimes (|0s_0\rangle \psi_0 - |1s_0\rangle \psi_1)}{2\sqrt{2}}
\end{aligned}$$

```

Step 1. Alice and Bob generates an entangled pair. Bob prepares a quantum state in a separate qubit.

```
In[4]:= qc1 = QuantumCircuit[LogicalForm[Ket[], S@{1, 2}],
ProductState[S[3] → {ψ[0], ψ[1]}, "Label" → Ket[ψ]], S[1, 6],
CNOT[S[1], S[2]], "Separator", "Invisible" → S[1.5]]

Out[4]=
```

Step 2. Bob performs a Bell measurement on his qubits. This can be done by reversing the entangler circuit.

```
In[5]:= qc2a = QuantumCircuit[CNOT[S[2], S[3]], S[2, 6]];
op = ExpressionFor[qc2a];
bs = BellState[S@{2, 3}];
ls = op ** bs;
LogicalForm@Transpose@Thread[bs >> ls] // TableForm

Out[5]/TableForm=

$$\frac{|0_{S_2}0_{S_3}\rangle + |1_{S_2}1_{S_3}\rangle}{\sqrt{2}} \rightarrow |0_{S_2}0_{S_3}\rangle$$


$$\frac{|0_{S_2}1_{S_3}\rangle + |1_{S_2}0_{S_3}\rangle}{\sqrt{2}} \rightarrow |0_{S_2}1_{S_3}\rangle$$

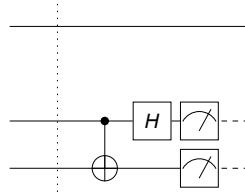

$$\frac{|0_{S_2}1_{S_3}\rangle - |1_{S_2}0_{S_3}\rangle}{\sqrt{2}} \rightarrow |1_{S_2}1_{S_3}\rangle$$


$$\frac{|0_{S_2}0_{S_3}\rangle - |1_{S_2}1_{S_3}\rangle}{\sqrt{2}} \rightarrow |1_{S_2}0_{S_3}\rangle$$

```

This is the corresponding quantum circuit model.

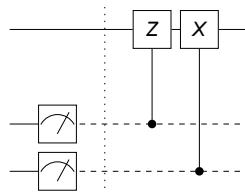
```
In[6]:= qc2 = QuantumCircuit["Separator", qc2a,
  Measurement@S@{2, 3}, "Visible" >> S[1], "Invisible" >> S[1.5]]
```



Step 3. Bob sends the measurement outcome to Alice through a classical channel.

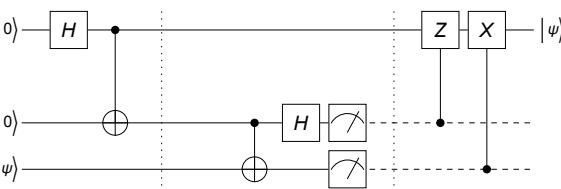
Step 4. Alice applies a proper unitary gate on her qubit in accordance with Bob's message.
These two steps are simulated by a feedback control.

```
In[7]:= qc2 = QuantumCircuit[Measurement@S@{2, 3}, "Separator",
  ControlledU[S[2], S[1, 3]], ControlledU[S[3], S[1, 1]], "Invisible" >> S[1.5]]
```



Combining all the steps, one can transmit a quantum state to a remote party without a quantum channel.

```
In[8]:= qc = QuantumCircuit[in = ProductState[S[3] >> {\psi[0], \psi[1]}, "Label" >> Ket[\psi]],
  LogicalForm[Ket[], S@{1, 2}], S[1, 6], CNOT[S[1], S[2]], "Separator",
  "Spacer", CNOT[S[2], S[3]], S[2, 6], Measurement[S@{2, 3}],
  "Separator", ControlledU[S[2], S[1, 3]], ControlledU[S[3], S[1, 1]],
  ProductState[S[1] >> {\psi[0], \psi[1]}, "Label" >> Ket[\psi]], "Invisible" >> S[1.5]]
```



Check the result.

```
In[7]:= out = ExpressionFor[qc] /. {Conjugate[\psi[0]] * \psi[0] + Conjugate[\psi[1]] * \psi[1] \rightarrow 1};
LogicalForm@QuissoFactor[out, S@{2, 3}]
Out[7]=  $|\theta_{S_2} 1_{S_3}\rangle \otimes (|\theta_{S_1}\rangle \psi_0 + |1_{S_1}\rangle \psi_1)$ 
```

Try with an explicit numbers as well.

```
In[8]:=  $\{\psi[0], \psi[1]\} = \text{Normalize}@RandomVector[];$ 
out = ExpressionFor[qc];
in \rightarrow LogicalForm@QuissoFactor[out, S@{2, 3}]
Out[8]=  $((0.510954 + 0.0179195 i) |0\rangle + (0.753843 - 0.412705 i) |1\rangle)_{S_3} \rightarrow$ 
 $|\theta_{S_2} \theta_{S_3}\rangle \otimes ((0.510954 + 0.0179195 i) |\theta_{S_1}\rangle + (0.753843 - 0.412705 i) |1_{S_1}\rangle)$ 
```

4.2 Deutch-Jozsa Algorithm & Variants

The Deutsch-Jozsa algorithm (Deutsch, 1985; Deutsch & Jozsa, 1992) is known to be the first quantum algorithm that is faster than the best classical counterpart. Although it is not useful for practical applications, it is still interesting as it illustrates some aspects of *quantum parallelism* providing a glimpse of a hint why quantum computers are faster than classical computers. It has inspired the two variants, the Bernstein-Vazirani algorithm, and Simon’s algorithm.

4.2.1 Quantum Oracle

An *oracle* in computer science is a “black box” operation with a certain unknown property. In a decision problem such as the Deutch-Jozsa and related problems, we are supposed to figure out the unknown property by running the oracle. Before going further, let us first examine a *quantum oracle*—a quantum mechanical implementation of an oracle.

Typically, a classical oracle is described by a binary function

$$f : \{0, 1\}^m \rightarrow \{0, 1\}^n, \quad (4.10)$$

which maps an m -bit input to an n -bit output. The function $f(x)$ may not be invertible in general.

Given a classical oracle f , a proper quantum implementation should operate on qubits and allow superposition in the input states. One naive approach is to define an operator \hat{O} by $\hat{O}|x\rangle = |f(x)\rangle$ for any state in the logical basis of the m -qubit register. It would not work because, as mentioned above, the function $f(x)$ is not invertible in general and hence the operator \hat{O} cannot be unitary.

To overcome such an issue, first extend the mapping f by adding an auxiliary register of n bits to the input and keeping the original input value so that both the

input and output registers have $(m+n)$ bits. The extended mapping, $\{0,1\}^{m+n} \rightarrow \{0,1\}^{m+n}$, is then defined by the association

$$(x, y) \mapsto (x, f(x) \oplus y), \quad (4.11)$$

where $x \in \{0,1\}^m$ and $y \in \{0,1\}^n$ are the bit strings of the m -bit native register and the n -bit auxiliary register, respectively. Although the function $f(x)$ itself may not be invertible, the extended mapping in (4.11) is always one-to-one regardless of the function $f(x)$ —Problem 4.1. Due to this property, the extended mapping in (4.11) is widely used to convert a classical code to a form that is suitable for (classical) *reversible computation*.^{4.3}

The *quantum oracle* corresponding to the classical oracle f is simply an implementation of the extended mapping (4.11) on quantum registers: It is a quantum gate operation defined by the association

$$\hat{U}_f : |x\rangle \otimes |y\rangle \mapsto |x\rangle \otimes |f(x) \oplus y\rangle, \quad (4.12)$$

where $|x\rangle$ and $|y\rangle$ are the logical basis states belonging to the native and auxiliary register of m and n qubits, respectively. As the extended mapping (4.11) is one-to-one and the logical basis states are orthonormal, the operator \hat{U}_f is unitary—Problem 4.1. It is important to recall that \hat{U}_f is a linear operator and can act on any arbitrary superposition states. In the quantum circuit model, a quantum oracle is depicted diagrammatically as following



In this particular example, the first three qubits are from the native register and the last two qubits belong to the auxiliary register. The bit values of the auxiliary qubits are flipped conditionally depending on the value $f(x)$ as a function of the bit values x of the native qubits.

Here is a simple example, which is used in the Deutsch-Jozsa algorithm.

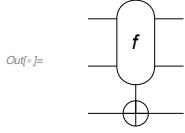
Consider a balanced function f as a classical oracle.

```
f[0, 0] = f[1, 1] = 0;
f[0, 1] = f[1, 0] = 1;
```

^{4.3}For a complete reversible computation, there is another important step required to remove the “garbage” bits.

Here is the corresponding quantum oracle.

```
In[1]:= cc = S@{1, 2};
tt = S[3];
qc = QuantumCircuit[Oracle[f, cc, tt]]
```



```
In[2]:= op = ExpressionFor[qc]
Out[2]=  $\frac{1}{2} + \frac{1}{2} S_1^z S_2^z - \frac{1}{2} S_1^z S_2^z S_3^x + \frac{S_3^x}{2}$ 

In[3]:= bs = Basis@Join[cc, {tt}];
bs // LogicalForm
Out[3]= \{ |0_{S_1} 0_{S_2} 0_{S_3}\rangle, |0_{S_1} 0_{S_2} 1_{S_3}\rangle, |0_{S_1} 1_{S_2} 0_{S_3}\rangle, |0_{S_1} 1_{S_2} 1_{S_3}\rangle, |1_{S_1} 0_{S_2} 0_{S_3}\rangle, |1_{S_1} 0_{S_2} 1_{S_3}\rangle, |1_{S_1} 1_{S_2} 0_{S_3}\rangle, |1_{S_1} 1_{S_2} 1_{S_3}\rangle\}

In[4]:= out = op ** bs;
out // LogicalForm
Out[4]= \{ |0_{S_1} 0_{S_2} 0_{S_3}\rangle, |0_{S_1} 0_{S_2} 1_{S_3}\rangle, |0_{S_1} 1_{S_2} 1_{S_3}\rangle, |0_{S_1} 1_{S_2} 0_{S_3}\rangle, |1_{S_1} 0_{S_2} 1_{S_3}\rangle, |1_{S_1} 0_{S_2} 0_{S_3}\rangle, |1_{S_1} 1_{S_2} 0_{S_3}\rangle, |1_{S_1} 1_{S_2} 1_{S_3}\rangle\}
```

```
In[5]:= Thread[bs \[Rule] out] // LogicalForm // TableForm
```

```
Out[5]//TableForm=
\begin{array}{l}
|0_{S_1} 0_{S_2} 0_{S_3}\rangle \rightarrow |0_{S_1} 0_{S_2} 0_{S_3}\rangle \\
|0_{S_1} 0_{S_2} 1_{S_3}\rangle \rightarrow |0_{S_1} 0_{S_2} 1_{S_3}\rangle \\
|0_{S_1} 1_{S_2} 0_{S_3}\rangle \rightarrow |0_{S_1} 1_{S_2} 1_{S_3}\rangle \\
|0_{S_1} 1_{S_2} 1_{S_3}\rangle \rightarrow |0_{S_1} 1_{S_2} 0_{S_3}\rangle \\
|1_{S_1} 0_{S_2} 0_{S_3}\rangle \rightarrow |1_{S_1} 0_{S_2} 1_{S_3}\rangle \\
|1_{S_1} 0_{S_2} 1_{S_3}\rangle \rightarrow |1_{S_1} 0_{S_2} 0_{S_3}\rangle \\
|1_{S_1} 1_{S_2} 0_{S_3}\rangle \rightarrow |1_{S_1} 1_{S_2} 0_{S_3}\rangle \\
|1_{S_1} 1_{S_2} 1_{S_3}\rangle \rightarrow |1_{S_1} 1_{S_2} 1_{S_3}\rangle
\end{array}
```

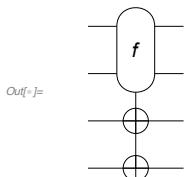
Let us consider another example, which is used in Simon's algorithm.

Consider a two-to-one function as a classical oracle.

```
f[0, 0] = f[1, 1] = {0, 1};
f[0, 1] = f[1, 0] = {1, 1};
```

Here is an implementation of the corresponding quantum oracle.

```
In[1]:= cc = S@{1, 2};
tt = S@{3, 4};
qc = QuantumCircuit[Oracle[f, cc, tt]]
```



```

In[=]:= op = Elaborate@QuissoOracle[f, cc, tt]
Out[=]=  $\frac{1}{2} S_3^x S_4^x + \frac{1}{2} S_1^z S_2^z S_4^x - \frac{1}{2} S_1^z S_2^z S_3^x S_4^x + \frac{S_4^x}{2}$ 

In[=]:= bs = Basis@Join[cc, tt];
bs // LogicalForm
Out[=]= { |0s10s20s30s4⟩, |0s10s21s30s4⟩, |0s10s21s31s4⟩,
          |0s11s20s30s4⟩, |0s11s20s31s4⟩, |0s11s21s30s4⟩, |0s11s21s31s4⟩,
          |1s10s20s30s4⟩, |1s10s20s31s4⟩, |1s10s21s30s4⟩, |1s10s21s31s4⟩,
          |1s11s20s30s4⟩, |1s11s20s31s4⟩, |1s11s21s30s4⟩, |1s11s21s31s4⟩}

In[=]:= out = op ** bs;
out // LogicalForm
Out[=]= { |0s10s20s31s4⟩, |0s10s20s30s4⟩, |0s10s21s31s4⟩, |0s10s21s30s4⟩,
          |0s11s21s31s4⟩, |0s11s21s30s4⟩, |0s11s20s31s4⟩, |0s11s20s30s4⟩,
          |1s10s21s31s4⟩, |1s10s21s30s4⟩, |1s10s20s31s4⟩, |1s10s20s30s4⟩,
          |1s11s20s31s4⟩, |1s11s20s30s4⟩, |1s11s21s31s4⟩, |1s11s21s30s4⟩}

In[=]:= Thread[Rule[bs, out]] // LogicalForm // TableForm
Out[=]//TableForm=
|0s10s20s30s4⟩ → |0s10s20s31s4⟩
|0s10s20s31s4⟩ → |0s10s20s30s4⟩
|0s10s21s30s4⟩ → |0s10s21s31s4⟩
|0s10s21s31s4⟩ → |0s10s21s30s4⟩
|0s11s20s30s4⟩ → |0s11s21s31s4⟩
|0s11s20s31s4⟩ → |0s11s21s30s4⟩
|0s11s21s30s4⟩ → |0s11s20s31s4⟩
|0s11s21s31s4⟩ → |0s11s20s30s4⟩
|1s10s20s30s4⟩ → |1s10s21s31s4⟩
|1s10s20s31s4⟩ → |1s10s21s30s4⟩
|1s10s21s30s4⟩ → |1s10s20s31s4⟩
|1s10s21s31s4⟩ → |1s10s20s30s4⟩
|1s11s20s30s4⟩ → |1s11s21s31s4⟩
|1s11s20s31s4⟩ → |1s11s21s30s4⟩
|1s11s21s30s4⟩ → |1s11s20s31s4⟩
|1s11s21s31s4⟩ → |1s11s20s30s4⟩

```

There are several interesting features of quantum oracle to be noticed immediately from the definition in (4.12). In Section 2.2.1, we noted that the CNOT gate makes a copy of the logical state of the control register to the target register when the latter is initially prepared in the state $|0\rangle$ —see Eq. (2.36). It is exploited in many applications, especially, for the generation of entanglement—see Eqs. (2.33) and (2.37). Quantum oracle has a similar property: However, the quantum oracle makes a copy of the image $|f(x)\rangle$ rather than the state $|x\rangle$ itself of the native register to the ancillary register,

$$|x\rangle \otimes |0\rangle \mapsto |x\rangle \otimes |f(x)\rangle . \quad (4.14)$$

Suppose that the native quantum register in the superposition $\frac{1}{2^{m/2}} \sum_{x=0}^{2^m-1} |x\rangle$ and the ancillary quantum register in the state $|0\rangle \equiv |0\rangle^{\otimes n}$. The quantum oracle

transforms the state as

$$\frac{1}{2^{m/2}} \sum_{x=0}^{2^m-1} |x\rangle \otimes |0\rangle \mapsto \frac{1}{2^{m/2}} \sum_{x=0}^{2^m-1} |x\rangle \otimes |f(x)\rangle . \quad (4.15)$$

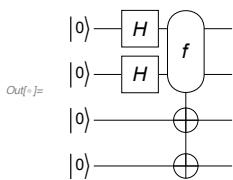
Just like the state in (2.33) from the CNOT gate, the state (4.15) from the quantum oracle is also entangled unless f is constant. In this case, the entanglement is controlled by the (classical) oracle f .

Here is demonstrate the feature of quantum oracle that makes a copy of the image $f(x)$ of the native register to the ancillary qubit.

In this particular example, we consider a one-to-one function, but it can be arbitrary (but not constant if an entanglement is necessary).

```
f[0, 0] = {1, 1};
f[0, 1] = {1, 0};
f[1, 0] = {0, 1};
f[1, 1] = {0, 0};
```

```
In[7]:= cc = {1, 2};
tt = {3, 4};
qc = QuantumCircuit[
  LogicalForm[Ket[], Join[S@cc, S@tt]], S[cc, 6], Oracle[f, S@cc, S@tt]]
```



The output state is an entangled state (unless the classical oracle f is constant).

```
In[8]:= out = ExpressionFor[qc];
LogicalForm[out, Join[S@cc, S@tt]]
Out[8]=  $\frac{1}{2} |\Theta_{S_1} \Theta_{S_2} 1_{S_3} 1_{S_4}\rangle + \frac{1}{2} |\Theta_{S_1} 1_{S_2} 1_{S_3} \Theta_{S_4}\rangle + \frac{1}{2} |1_{S_1} \Theta_{S_2} \Theta_{S_3} 1_{S_4}\rangle + \frac{1}{2} |1_{S_1} 1_{S_2} \Theta_{S_3} \Theta_{S_4}\rangle$ 
```

To make clearer the copies made tot the ancillary register, it may be useful to rewrite the state vector in a form that distinguishes the native and ancillary register.

```
In[9]:= QuissoFactor[out, S@cc] // LogicalForm
Out[9]=  $|\Theta_{S_1} \Theta_{S_2}\rangle \otimes \left(\frac{1}{2} |1_{S_3} 1_{S_4}\rangle\right) + |\Theta_{S_1} 1_{S_2}\rangle \otimes \left(\frac{1}{2} |1_{S_3} \Theta_{S_4}\rangle\right) + |1_{S_1} \Theta_{S_2}\rangle \otimes \left(\frac{1}{2} |0_{S_3} 1_{S_4}\rangle\right) + |1_{S_1} 1_{S_2}\rangle \otimes \left(\frac{1}{2} |0_{S_3} \Theta_{S_4}\rangle\right)$ 
```

In Section 2.2.2, we have seen that the controlled- U gate induces a phase shift on the control register—rather than on the target register—when the target register is in an eigenstate of the unitary operator. Similar method can be used to induce a phase shift conditionally on every terms that satisfy a certain condition.

For example, let $f : \{0, 1\}^n \rightarrow \{0, 1\}$ is a classical oracle, and suppose that we are given a state

$$|\psi\rangle = \sum_{x=0}^{2^n-1} |x\rangle \psi_x \quad (4.16)$$

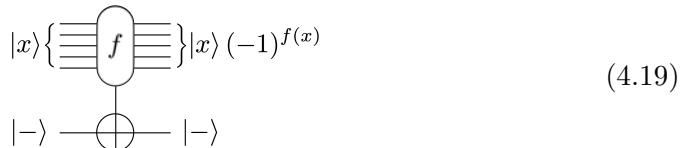
on an n -qubit register. We want to get a new state where every term $|x\rangle$ satisfying $f(x) = 1$ flips its amplitude ψ_x to $-\psi_x$

$$|\psi'\rangle = \sum_{x=0}^{2^n-1} |x\rangle (-1)^{f(x)} \psi_x \quad (4.17)$$

In other words, we want an *effective* quantum gate that maps the logical basis states as

$$|x\rangle \mapsto |x\rangle (-1)^{f(x)}, \quad x = 0, 1, 2, \dots, 2^n - 1. \quad (4.18)$$

This can be achieved using the quantum oracle corresponding to the function f and preparing an auxiliary qubit in the state $|-\rangle := (|0\rangle - |1\rangle)/\sqrt{2}$, the eigenstate of the Pauli X operator belonging to the eigenvalue -1 , as depicted in the quantum circuit model



Indeed, for a logical sitate $|x\rangle$ with $f(x) = 1$,

$$|x\rangle \otimes |-\rangle = \frac{|x\rangle \otimes |0\rangle - |x\rangle \otimes |1\rangle}{\sqrt{2}} \mapsto \frac{|x\rangle \otimes |1\rangle - |x\rangle \otimes |0\rangle}{\sqrt{2}} = -|x\rangle \otimes |-\rangle, \quad (4.20)$$

while nothing happens for $|x\rangle$ with $f(x) = 0$. One can find an example of more general conditional phase shift in Problem 4.2.

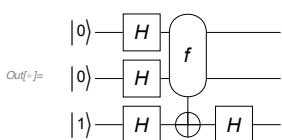
This is an interesting twist of the quantum oracle. It is useful in Grover's search algorithm.

The classical oracle mark the solutions, $\{0, 0\}=0$ and $\{1, 1\}=1$ in this particular case.

`f[0, 0] = 1; f[0, 1] = 0; f[1, 0] = 0; f[1, 1] = 1;`

This quantum circuit model implements the desired mapping.

```
In[5]:= cc = {1, 2};
tt = {3};
ct = Join[cc, tt];
qc = QuantumCircuit[LogicalForm[Ket[S[tt] \[Rule] 1], S[ct]],
S[ct, 6], Oracle[f, S@cc, S@tt], S[tt, 6]]
```



```
In[5]:= out = ExpressionFor[qc];
QuissoFactor[out, S[tt]] // LogicalForm
Out[5]=  $\left|1_{S_3}\right\rangle \otimes \left(-\frac{1}{2} \left|\Theta_{S_1} 0_{S_2}\right\rangle + \frac{1}{2} \left|\Theta_{S_1} 1_{S_2}\right\rangle + \frac{1}{2} \left|1_{S_1} \Theta_{S_2}\right\rangle - \frac{1}{2} \left|1_{S_1} 1_{S_2}\right\rangle\right)$ 
```

Check the result.

```
In[6]:= bs = Basis[S@cc];
ff = f@@@IntegerDigits[Range[0, 2^2 - 1], 2, 2];
Thread[bs → Power[-1, ff]] // LogicalForm
Out[6]=  $\{\left|\Theta_{S_1} 0_{S_2}\right\rangle \rightarrow -1, \left|\Theta_{S_1} 1_{S_2}\right\rangle \rightarrow 1, \left|1_{S_1} \Theta_{S_2}\right\rangle \rightarrow 1, \left|1_{S_1} 1_{S_2}\right\rangle \rightarrow -1\}$ 
```

```
In[7]:= new = bs.Power[-1, ff] / 2;
new // LogicalForm
Out[7]=  $\frac{1}{2} \left(-\left|\Theta_{S_1} 0_{S_2}\right\rangle + \left|\Theta_{S_1} 1_{S_2}\right\rangle + \left|1_{S_1} \Theta_{S_2}\right\rangle - \left|1_{S_1} 1_{S_2}\right\rangle\right)$ 
```

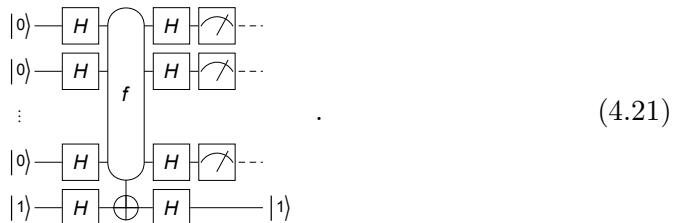
4.2.2 Deutsch-Jozsa Algorithm

Let us now discuss the Deutsch-Jozsa algorithm. Proposed first by [Deutsch & Jozsa \(1992\)](#), the algorithm determines whether a given function is balanced or constant. It is of little use in practice, but it is one of the first examples of a quantum algorithm that is exponentially faster than any possible classical algorithms. Further, it is so simple that it reveals clearly some aspects of *quantum parallelism* and the operational principle of quantum oracle.

The Deutsch-Jozsa problem is defined as following: Consider a classical function $f : \{0, 1\}^n \rightarrow \{0, 1\}$. It is a classical oracle taking n -bit inputs and returning 0 or 1. It is known that the function is either constant—either 0 or 1 for all inputs—or balanced—0 for one half of the possible inputs and 1 for the other half. The task is to determine whether f is constant or balanced by using the oracle the least number of times.

In classical algorithms, it is known that one has to evaluate the oracle $(2^{n-1} + 1) \approx 2^n/2$ times in the worst case. That is, one has to try almost half of possible inputs before reliably determining the unknown property of the oracle. As we will see now, the Deutsch-Jozsa algorithm figures out the property from a single query to the *quantum oracle*.

The Deutsch-Jozsa algorithm is summarized in the quantum circuit model



The last qubit is an ancillary qubit to induce the conditional phase shifts on the first n qubits. The Hadamard gate on it ensures that the ancillary qubit is in the

eigenstate $|-\rangle$ of the Pauli X operator just before the quantum oracle operates. According to Eq. (2.18), the Hadamard gates before the quantum oracle create the linear superposition of all the logical basis states of the n -qubit native register (normalization ignored)

$$|0\rangle \xrightarrow{\hat{H}^{\otimes n}} \frac{1}{2^{n/2}} \sum_{x=0}^{2^n-1} |x\rangle \quad (4.22)$$

Next, as seen in Eq. (4.19), the quantum oracle induces the conditional phase shifts

$$\sum_{x=0}^{2^n-1} |x\rangle \rightarrow \sum_x |x\rangle (-1)^{f(x)} \quad (4.23)$$

Another operation of the Hadamard gates shuffles and causes additional phase factors in accordance with (2.19), and leads to the output state to be measured

$$\sum_x |x\rangle (-1)^{f(x)} \xrightarrow{\hat{H}^{\otimes n}} \frac{1}{2^n} \sum_{y=0}^{2^n-1} |y\rangle \sum_{x=0}^{2^n-1} (-1)^{f(x)+x \cdot y} \quad (4.24)$$

To see the effect of the function f on the final state, suppose that f is a constant function. Then, the output state is given by

$$\frac{(-1)^{f(0)}}{2^n} \sum_{y=0}^{2^n-1} |y\rangle \sum_{x=0}^{2^n-1} (-1)^{x \cdot y} = (-1)^{f(0)} |0\rangle \quad (4.25)$$

Every measurement on each of the n qubits should yield zero with unit probability. To make the analysis more explicit, consider the probability to find the n -qubit register in the state $|0\rangle \equiv |0\rangle \otimes |0\rangle \otimes \cdots \otimes |0\rangle$

$$P_0 = \frac{1}{2^n} \left| \sum_{x=0}^{2^n-1} (-1)^{f(x)} \right|^2. \quad (4.26)$$

It assures that $P_0 = 1$ for a constant function f . When the function f is balanced, there are as many terms with -1 as 1 , and the sum is always zero. There must be at least one qubit in the state $|1\rangle$ if the function f is balanced. Therefore, to determine whether a given function f is constant or balanced, one needs to run the quantum oracle just once, and check if the measurement outcome is 0. If the outcome is 0, then the function must be constant; and balanced otherwise.

Consider a balanced function as an example.

```
f[0, 0] = f[1, 1] = 0;
f[0, 1] = f[1, 0] = 1;
```

Here is a quantum circuit model of the Deutsch-Jozsa algorithm. The final Hadamard gate on the third qubit is not necessary, but we put it here to make the output state more readable.

```
In[=]:= cc = {1, 2};
tt = {3};
all = {1, 2, 3};
qc = QuantumCircuit[LogicalForm[Ket[S[3] → 1], S@all],
S@all, 6], Oracle[f, S@cc, S@tt], S@all, 6]

Out[=]=
```

```
In[=]:= out = ExpressionFor[qc]
Out[=]= |1s11s21s3>
```

4.2.3 Bernstein-Vazirani Algorithm

Suppose that we are given a binary function $f : \{0, 1\}^n \rightarrow \{0, 1\}$, the value of which is determined by a secrete string s of n bits

$$f : x \mapsto x \cdot s \pmod{2}. \quad (4.27)$$

The Bernstein-Vazirani problem is to find the secrete bit string s by making queries to the oracle f . Classically, one needs n queries to the function f to infer the secrete string s .

The Bernstein-Vazirani algorithm [Bernstein & Vazirani \(1993, 1997\)](#) is implemented in the same quantum circuit model (4.21) as the Deutsch-Jozsa algorithm. Only the analysis of the final readout is different. The first set of the Hadamard gates and the quantum oracle in (4.21) leads the n -qubit register to the state

$$\sum_{x=0}^{2^n-1} |x\rangle (-1)^{x \cdot s}. \quad (4.28)$$

Taking the inverse of the mapping in (2.19), one can observe that the second set of the Hadamard gates converts the above state to the final state $|s\rangle$ set by the secrete bit string. Therefore, a simple measurement of the final state in the logical basis will just reveal the secrete string.

Consider a secrete string of bits. The task is to find the secrete string.

```
string = {0, 1};
```

In the Bernstein-Vazirani algorithm, the value of the classical oracle f is determined by the given secrete string.

```
Clear[f];
f[x_] := Mod[{x}.string, 2]
```

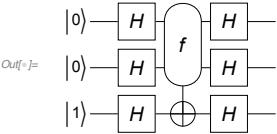
For example, $f[1,1]=0 \cdot 1 + 1 \cdot 1=1$.

In[^v] := **f**[1, 1]

Out[^v] = 1

Here is a quantum circuit model of the Bernstein-Vazirani algorithm. The final Hadamard gate on the third qubit is not necessary and we put it here to make the output state more readable.

```
In[v] := cc = {1, 2};
tt = {3};
all = {1, 2, 3};
qc = QuantumCircuit[LogicalForm[Ket[S[3] → 1], S@all],
S[all, 6], Oracle[f, S@cc, S@tt], S[all, 6]]
```



In[^v] := **out** = ExpressionFor[**qc**];

LogicalForm[**out**, S@cc]

Out[^v] = $|0_{S_1}1_{S_2}1_{S_3}\rangle$

Here is the secrete string successfully retrieved.

In[^v] := **answer** = **out**[S@cc]

Out[^v] = {0, 1}

4.2.4 Simon's Algorithm

Finally, let us turn to Simon's algorithm (Simon, 1997). It was the first quantum algorithm featuring an exponential speed-up over the best known classical algorithm for a specific problem. The algorithm is known to have inspired quantum factorization algorithm. Furthermore, it has recently been shown that Simon's algorithm can be used to break the symmetric-key cryptosystems.

In Simon's problem, we are given a function $f : \{0, 1\}^n \rightarrow \{0, 1\}^n$ and a secrete string s of n bits. It is known that for all $x, y \in \{0, 1\}^n$, $f(x) = f(y)$ if and only if $y = x \oplus s$. Note that f is either one-to-one ($s = 0$) or two-to-one ($s \neq 0$). The task is to find the secrete string s with as few queries to the function f as possible. Classically, one needs queries to $f(x)$ with up to $2^{n-1} + 1$ different inputs.^{4.4} Unlike the Deutsch-Jozsa problem, Simon's problem is known to be hard to solve even probabilistically.

Simon's algorithm is summarized in the two slightly different quantum circuit models in Fig. 4.1. The measurement on the ancillary (second) register in the quantum circuit model in Fig. 4.1 (b) can be delayed until the measurement on the native (first) register, or even dropped off completely as in the quantum circuit model in Fig. 4.1 (a). That is, it is not essential, but depending on your taste, it may simplify the analysys of the algorithm. Here we will adopt the quantum circuit

^{4.4}More precisely, one needs order of $\sqrt{2^n}$ queries to encounter a pair of two bit strings leading to the same result with probability greater than $1/2$. The number being $\sqrt{2^n}$ rather than 2^n is related to the so-called “birthday paradox”.

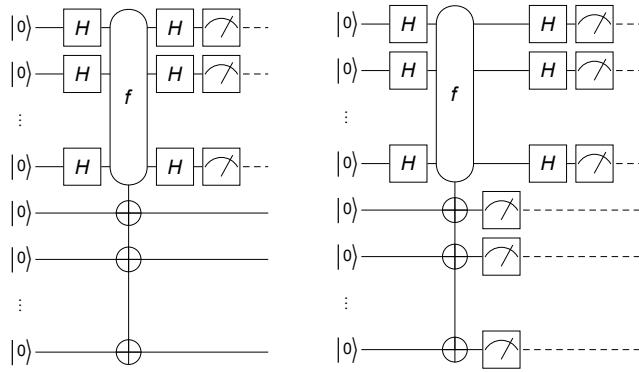


Figure 4.1: Quantum circuit models for Simon’s algorithm. The measurement on the ancillary register in (b) is not essential. It can be performed later than the second Hadamard gate on the native register, or even dropped off completely as in (a).

model in Fig. 4.1 (a). The first Hadamard gate on the native register transforms the input state of the whole system—again, see Eq. (2.18)—as

$$|0\rangle \otimes |0\rangle \xrightarrow{\hat{H}^{\otimes n}} \frac{1}{2^{n/2}} \sum_{x=0}^{2^n-1} |x\rangle \otimes |0\rangle . \quad (4.29)$$

As we noted in (4.15), the quantum oracle makes a copy of the image $|f(x)\rangle$ of the state $|x\rangle$ of the native register to the ancillary register, and leads to

$$\frac{1}{2^{n/2}} \sum_{x=0}^{2^n-1} |x\rangle \otimes |f(x)\rangle . \quad (4.30)$$

Finally, the second set of Hadamard gates on the native register—see Eq. (2.19)—maps the above state into

$$\sum_{y=0}^{2^n-1} |y\rangle \otimes \frac{1}{2^n} \sum_{x=0}^{2^n-1} |f(x)\rangle (-1)^{x \cdot y} . \quad (4.31)$$

The measurement on the native register yields an n -bit string y . The probability for a particular string y is determined by the squared norm, $P_y = \langle \psi_y | \psi_y \rangle$, of the y -dependent state $|\psi_y\rangle$ of the ancillary register

$$|\psi_y\rangle := \frac{1}{2^n} \sum_{x=0}^{2^n-1} |f(x)\rangle (-1)^{x \cdot y} . \quad (4.32)$$

Let us examine different cases: First, suppose that the secret string $s = 0$. In this case, the function f is one-to-one, and the state $|\psi_y\rangle$ in (4.32) simply consists

of terms that is a rearrangement of logical basis states

$$|\psi_y\rangle := \frac{1}{2^n} \sum_{z=0}^{2^n-1} |z\rangle (-1)^{y \cdot f^{-1}(z)}, \quad (4.33)$$

where f^{-1} is the inverse of f , and hence $P_y = 2^{-n}$ for all y . In other words, for $s = 0$, the measurement produces a random bit string y with uniform probability. Now, suppose that $s \neq 0$. In this case, the function f is two-to-one such that $f(x) = f(x \oplus s)$. The terms in the summation in (4.32) appear in pairs giving the same state, $|f(x)\rangle = |f(x \oplus s)\rangle$. To be more explicit, let $\mathcal{B} := f(\{0, 1\}^n)$ be the image of f . Then,

$$|\psi_y\rangle := \frac{1}{2^n} \sum_{z \in \mathcal{B}} |z\rangle \left\{ (-1)^{y \cdot a_z} + (-1)^{y \cdot b_z} \right\}, \quad (4.34)$$

where a_z and b_z are the elements in the preimage (or inverse image) of z under f , $f(a_z) = f(b_z) = z$. Since $b_z = a_z \oplus s$ and $(a_z \oplus s) \cdot y = (a_z \cdot y) \oplus (s \cdot y)$, it follows that

$$\left\{ (-1)^{a_z \cdot y} + (-1)^{b_z \cdot y} \right\} = (-1)^{y \cdot a_z} \{1 + (-1)^{y \cdot s}\}. \quad (4.35)$$

If $y \cdot s$ is odd, $y \cdot s = 1 \pmod{2}$, then $|\psi_y\rangle$ is a null vector, and $P_y = 0$ for such a bit string y . On the other hand, if $y \cdot s$ is even, $y \cdot s = 0 \pmod{2}$, then $|\psi_y\rangle$ is a finite vector and independent of y . That is, $P_y = 2^{1-n}$ regardless of y as long as $y \cdot s$ is even.

In short, the bit string y resulting from the measurement on the native (first) register always satisfy $y \cdot s = 0 \pmod{2}$ for any secret bit string s . To find $s \equiv (s_1 s_2 \dots s_n)_2$, one needs to run the algorithms repeated to get $(n - 1)$ linearly independent bit strings $y^{(i)} \equiv (y_1^{(i)} y_2^{(i)} \dots y_n^{(i)})_2$ ($i = 1, 2, \dots, n - 1$), and then solve the set of equations

$$\sum_{j=1}^n y_j^{(i)} s_j = 0 \pmod{2}. \quad (4.36)$$

It can be argued that the probability to get $n - 1$ linearly independent bit strings out of n runs is slightly larger than $1/4$. Therefore, the required number of queries to the quantum oracle is order of n , exponentially smaller than the classical algorithm.

Consider again a secret bit string.

string = {1, 1};

Consider a two-to-one function obeying the rule (specified in Simon's problem).

```
f[0, 0] = f[1, 1] = {0, 1};
f[0, 1] = f[1, 0] = {1, 1};
```

Here is an implementation of the corresponding quantum oracle.

```
In[4]:= cc = {1, 2};
tt = {3, 4};
all = Join[cc, tt];
qc = QuantumCircuit[LogicalForm[Ket[], S@all],
S[cc, 6], Oracle[f, S@cc, S@tt], S[cc, 6], Measurement[S@cc]]
```

```
Out[4]=
```

$$\frac{|1_{s_1}1_{s_2}0_{s_3}1_{s_4}\rangle}{\sqrt{2}} - \frac{|1_{s_1}1_{s_2}1_{s_3}1_{s_4}\rangle}{\sqrt{2}}$$

```
Out[4]= {1, 1}
```

```
In[5]:= eqs = Table[out = ExpressionFor[Matrix[qc], S@all];
result = Readout[out, S@cc], {2}]
```

```
Out[5]= {{1, 1}, {0, 0}}
```

Now let us examine a larger system. Suppose that we are given a secret bit string.

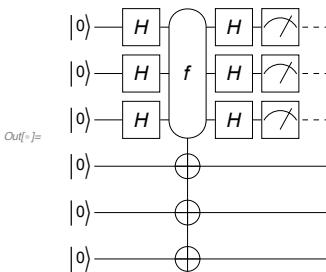
```
string = {1, 1, 0};
```

This is a function consistent with the above secret bit string.

```
f[0, 0, 0] = f[1, 1, 0] = {0, 1, 1};
f[0, 0, 1] = f[1, 1, 1] = {1, 1, 1};
f[0, 1, 0] = f[1, 0, 0] = {1, 0, 0};
f[0, 1, 1] = f[1, 0, 1] = {0, 0, 1};
```

Here is an implementation of the corresponding quantum oracle.

```
In[6]:= cc = {1, 2, 3};
tt = {4, 5, 6};
all = Join[cc, tt];
qc1 = QuantumCircuit[LogicalForm[Ket[], S@all],
S[cc, 6], Oracle[f, S@cc, S@tt], S[cc, 6]];
qc2 = QuantumCircuit[qc1, Measurement[S@cc]]
```



This is one way to get the measurement outcome.

```
In[5]:= out = ExpressionFor[qc2];
LogicalForm[out, S@all]
result = Readout[out, S@cc]
Out[5]= -  $\frac{1}{2} |0_{S_1} 0_{S_2} 1_{S_3} 0_{S_4} 0_{S_5} 1_{S_6}\rangle + \frac{1}{2} |0_{S_1} 0_{S_2} 1_{S_3} 0_{S_4} 1_{S_5} 1_{S_6}\rangle +$ 
 $\frac{1}{2} |0_{S_1} 0_{S_2} 1_{S_3} 1_{S_4} 0_{S_5} 0_{S_6}\rangle - \frac{1}{2} |0_{S_1} 0_{S_2} 1_{S_3} 1_{S_4} 1_{S_5} 1_{S_6}\rangle$ 
Out[5]= {0, 0, 1}
```

To make repeated measurements, it is more efficient to first compute the state just before the measurement.

```
new = ExpressionFor[qc1];
```

Now we perform the measurement repeatedly.

```
In[6]:= data = Table[out = Measurement[new, S@cc];
Readout[out, S@cc], {12}];
data // TableForm
Out[6]/TableForm=

$$\begin{array}{ccc} 1 & 1 & 1 \\ 1 & 1 & 1 \\ 0 & 0 & 1 \\ 1 & 1 & 1 \\ 0 & 0 & 1 \\ 1 & 1 & 1 \\ 0 & 0 & 1 \\ 0 & 0 & 0 \\ 1 & 1 & 0 \\ 1 & 1 & 0 \\ 0 & 0 & 1 \\ 0 & 0 & 1 \end{array}$$

```

As two linearly independent vectors (bit strings), we choose these:

```
In[7]:= mat = {{1, 1, 0}, {0, 0, 1}}
Out[7]= {{1, 1, 0}, {0, 0, 1}}
```

Then, the linear equation, `mat.ss=0 (mod 2)`, for the Boolean variables `ss:=s1,s2,s3` is given by the following, which agrees with the given secret bit string.

```
In[8]:= ss = {1, 1, 0}
Out[8]= {1, 1, 0}
In[9]:= Mod[mat.ss, 2]
Out[9]= {0, 0}
```

4.3 Quantum Fourier Transform (QFT)

The quantum Fourier transform is a unitary transformation of quantum states. It is analogous to the *discrete Fourier transform* of a finite set of numbers. In the case of quantum Fourier transform, the numbers are replaced with quantum states.

The quantum Fourier transform on a quantum computer consisting of n qubits can be performed efficiently with only $\mathcal{O}(n^2)$ elementary quantum gate operations,

compared to the $\mathcal{O}(n2^n)$ gates for the best known classical algorithm of discrete Fourier transform. This exponential speedup of the quantum Fourier transform algorithm compared with the classical counterpart enabled the celebrated quantum factorization algorithm to achieve a similar efficiency. Apart from the quantum factorization algorithm, the quantum Fourier transformation is a key part of many other quantum algorithms such as the quantum phase estimation (Section 4.4), the order-finding problem, the discrete logarithm, and most importantly the hidden subgroup problem. Such a wide range of applications of the quantum Fourier transform stem from the fact that all known quantum algorithms featuring exponential speedup over classical algorithms are variations of the hidden subgroup problem and, as first realized by Kitaev (1996), the key step to solve the latter problem is the quantum Fourier transform.

4.3.1 Definition and Physical Meaning

To make the physical meaning underlying the quantum Fourier transform, in this section we will take a slightly different notation for the logical basis states, and denote them by

$$|X_x\rangle := |x_1\rangle \otimes |x_2\rangle \otimes \cdots \otimes |x_n\rangle, \quad x = 0, 1, 2, \dots, 2^n - 1, \quad (4.37)$$

where as usual, x_j are the binary digits of the index $x = (x_1 x_2 \dots x_n)_2$. The quantum Fourier transform (QFT for short) is a unitary transformation defined by the association

$$\hat{U}_{\text{QFT}} |X_y\rangle := \frac{1}{2^{n/2}} \sum_{x=0}^{2^n-1} |X_x\rangle e^{ixp_y} \quad (y = 0, 1, 2, \dots, 2^n - 1), \quad (4.38)$$

where p_y is the y th “wave number” defined by

$$p_y := \frac{2\pi y}{2^n} = 2\pi(0.y_1 y_2 \dots y_n)_2. \quad (4.39)$$

The expression in the right-hand side of (4.38) is of familiar form of discrete Fourier transform except that instead of numerical coefficients to the exponential factor e^{ixp_y} there appear state vectors. Therefore, one may think of the quantum Fourier transform as a *vector-valued* discrete Fourier transformation.

In many physical applications, however, there is a more important aspect of the quantum Fourier transform than the formal similarity to the discrete Fourier transform: It turns out to be useful and inspiring to regard the QFT as a basis change, a unitary transformation from the logical basis

$$\{|X_x\rangle : x = 0, 1, \dots, 2^n - 1\} \quad (4.40)$$

to the so-called “conjugate basis”

$$\{|P_y\rangle := \hat{U}_{\text{QFT}} |X_y\rangle : y = 0, 1, \dots, 2^n - 1\}. \quad (4.41)$$

The key observation is that in the “continuum” limit ($L := 2^n \rightarrow \infty$), the logical basis corresponds to the eigenbasis of “position” and the conjugate basis to that of “momentum”.^{4.5} Indeed, one can see that the following two observables

$$\hat{X} := \sum_x |X_x\rangle x \langle X_x|, \quad \hat{P} := \sum_y |P_y\rangle p_y \langle P_y| \quad (4.42)$$

bearing eigenvalues x and p_y , respectively, satisfy the relation

$$e^{-ia\hat{P}} \hat{X} e^{+ia\hat{P}} = \hat{X} - a. \quad (4.43)$$

It implies that $e^{-ia\hat{P}}$ is a translation operator, and \hat{P} is the generator of translation, i.e., the momentum.

Quantum Fourier transform appears frequently in many quantum algorithms (notably the quantum phase estimation algorithm in Section 4.4), either in the explicit or disguised form. The relation between the logical and conjugate basis defined above is useful to understand the principle behind the particular algorithms.

4.3.2 Quantum Implementation

The quantum Fourier transform (QFT) algorithm is to efficiently implement the unitary operator (4.37) in terms of elementary gates.

Getting back to the definition, the quantum Fourier transform maps the quantum states by

$$\hat{U}_{\text{QFT}} |X_y\rangle = \frac{1}{2^{n/2}} \sum_{x=0}^{2^n-1} |X_x\rangle e^{ixp_y} \quad (4.44)$$

with the wave number

$$p_y := \frac{2\pi y}{2^n} = 2\pi(0.y_1y_2\dots y_n)_2. \quad (4.45)$$

As the characteristic properties of the Fourier transform of any kind are attributed to the oscillatory phase factor e^{ixp_y} , we start with elaborating it. The product xp_y in the exponent is given by

$$xp_y = \frac{2\pi xy}{2^n} = 2\pi(0.x_1x_2\dots x_n)_2 y = 2\pi y \sum_{k=1}^n x_k 2^{-k}, \quad (4.46)$$

which recasts the phase factor into the form

$$e^{ixp_y} = \prod_{k=1}^n e^{2\pi iyx_k 2^{-k}}. \quad (4.47)$$

^{4.5}This notion can be made rigorous even in a finite dimensional Hilbert space using the unitary operators $\exp(-ip\hat{X})$ and $\exp(-ia\hat{P})$, which generate the so-called *Weyl-Heisenberg group*. See Weyl (1931, Section IV.14) for example.

Putting it back in (4.44) and rewriting the sum over the integer values x with the sum over the bit values $x_j \in \{0, 1\}$ lead to

$$\hat{U}_{\text{QFT}} |X_y\rangle = \bigotimes_{k=1}^n \sum_{x_k} |x_k\rangle e^{2\pi i x_k y / 2^k}. \quad (4.48)$$

For a fixed k , the ratio $y/2^k$ is represented in the binary digits as

$$y/2^k = (y_1 y_2 \cdots y_{n-k} \cdot y_{n-k+1} \cdots y_{n-1} y_n)_2. \quad (4.49)$$

The integer part does not affect the sum because $e^{2\pi i m} = 1$ for any integer m ; only the fractional part $(0.y_{n-k+1} \cdots y_{n-1} y_n)_2$ does. Depriving the phase factors $e^{ix_k p_y}$ of the integer parts, we arrive at the expression for the result of the transformation

$$\hat{U}_{\text{QFT}} |X_y\rangle = \bigotimes_{k=1}^n \sum_{x_k} |x_k\rangle e^{2\pi i x_k (0.y_{n-k+1} \cdots y_{n-1} y_n)_2}. \quad (4.50)$$

Explicitly writing the sums over bit values x_j , it reads as

$$\begin{aligned} \hat{U}_{\text{QFT}} |X_y\rangle &= \left(\frac{|0\rangle + |1\rangle e^{2\pi i 0.y_n}}{\sqrt{2}} \right) \otimes \left(\frac{|0\rangle + |1\rangle e^{2\pi i 0.y_{n-1} y_n}}{\sqrt{2}} \right) \otimes \\ &\quad \cdots \otimes \left(\frac{|0\rangle + |1\rangle e^{2\pi i 0.y_1 y_2 \cdots y_n}}{\sqrt{2}} \right). \end{aligned} \quad (4.51)$$

Interestingly, the state resulting from the quantum Fourier transform is clearly a product state. The first factor is stored in the first qubit of the quantum register, and consecutive factors in the corresponding qubits in the same order. For the later analysis, it is useful to reverse the order. It is equivalent to perform the qubit-reversing unitary transformation

$$\hat{U}_{\text{QBR}} |y_1\rangle \otimes |y_2\rangle \otimes \cdots \otimes |y_n\rangle = |y_n\rangle \otimes \cdots \otimes |y_2\rangle \otimes |y_1\rangle. \quad (4.52)$$

Applying first the qubit-resersing transformation before the quantum Fourier transform, we arrive at the final expression that we analyse subsequently

$$\begin{aligned} \hat{U}_{\text{QFT}} \hat{U}_{\text{QBR}} |X_y\rangle &= \left(\frac{|0\rangle + |1\rangle e^{2\pi i 0.y_1}}{\sqrt{2}} \right) \otimes \left(\frac{|0\rangle + |1\rangle e^{2\pi i 0.y_2 y_1}}{\sqrt{2}} \right) \otimes \\ &\quad \cdots \otimes \left(\frac{|0\rangle + |1\rangle e^{2\pi i 0.y_n \cdots y_2 y_1}}{\sqrt{2}} \right). \end{aligned} \quad (4.53)$$

So far, we have done nothing but re-expressing the defining equation (4.44) of the quantum Fourier transform. Now we want to identify the elementary quantum logic gates that compose the transform. The product for in (4.53) is particularly useful to analyse: Let us examine the state stored on the k th qubit,

$$\frac{|0\rangle + |1\rangle e^{2\pi i (0.y_k \cdots y_2 y_1)_2}}{\sqrt{2}}. \quad (4.54)$$

Noting that

$$2\pi(0.y_k \dots y_2 y_1)_2 = \sum_{j=1}^k y_j 2^{j-k}\pi, \quad (4.55)$$

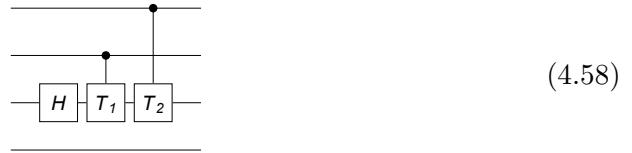
we see that the qubits $1, 2, \dots, k$ makes relative phase shifts $2^{1-k}\pi, 2^{2-k}\pi, \dots, \pi$, respectively, on the state in (4.54). Among these, the phase shift π depends on the state $|y_k\rangle$ of the k th qubit itself, and is equivalent to the Hadamard gate. Therefore, we can regard that the state (4.54) stored on the k th qubit is resulting from the transformation

$$|y_1\rangle \otimes \dots \otimes |y_{k-1}\rangle \otimes |y_k\rangle \mapsto |y_1\rangle \otimes \dots \otimes |y_{k-1}\rangle \otimes \prod_{j=1}^{k-1} \hat{T}^{y_j}(2^{j-k}\pi) \hat{H} |y_k\rangle, \quad (4.56)$$

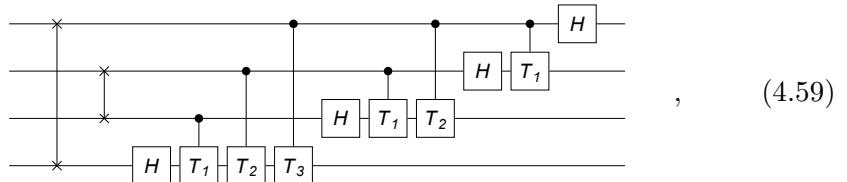
where $\hat{T}(\phi)$ denotes the relative phase shift

$$\hat{T}(\phi) = |0\rangle\langle 0| + |1\rangle\langle 1| e^{i\phi} \doteq \begin{bmatrix} 1 & 0 \\ 0 & e^{i\phi} \end{bmatrix} \quad (4.57)$$

As the actual operation of $\hat{T}^{y_j}(2^{j-k}\pi)$ is determined by the logical state $|y_j\rangle$ of the j th qubit, it is a controlled unitary gate. For example, the state store on the third qubit is represented in the quantum circuit model as

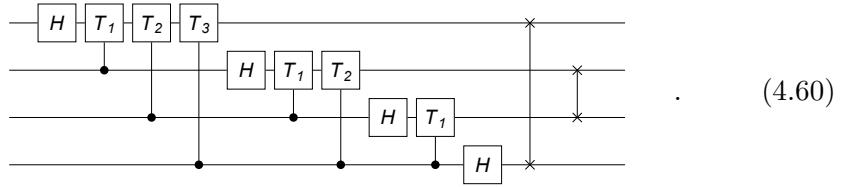


where we have used the short-hand notation $\hat{T}_l := \hat{T}(\pi/2^l)$. Overall, the quantum Fourier transformation can be represented in the quantum circuit model as



where we have implemented the qubit-reversing transformation \hat{U}_{QBT} by applying the SWAP gates on the pairs of qubits from the first and second half of the quantum register. In the above quantum circuit model, we have applied \hat{U}_{QBR} at the begining. One can apply it at the end as well: As \hat{U}_{QBR} corresponds to simply reversing the order of qubits, the quantum circuit model in (4.59) is identical to

the following model



This quantum circuit model for the quantum Fourier transformation is found more commonly in the literature.

Here we simulate the quantum Fourier transform on a quantum register of 4 qubits.

```
$n = 4;
```

Here defined are the controlled-phase gates.

```
CP[j_, j_] := S[j, 6]
CP[j_, k_] := ControlledU[S[j],
    Phase[Pi Power[2, k - j], S[k], "Label" → Subscript["T", j - k]]
] /; j > k
CP[j_, k_] := ControlledU[S[j],
    Phase[Pi Power[2, j - k], S[k], "Label" → Subscript["T", k - j]]
] /; j < k
```

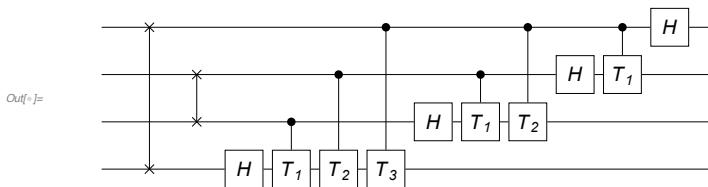
We denote by the label T_k the relative phase shift by $\pi/2^k$: $T_k := \begin{pmatrix} 1 & 0 \\ 0 & e^{i\pi/2^k} \end{pmatrix}$.

This is just a short-hand function.

```
SW[j_] := SWAP[S[j], S[$n - j + 1]]
SW[All] := Table[SW[j], {j, 1, $n / 2}]
```

This is the standard implementation, which can be seen commonly in textbooks.

```
In[7]:= gates = Flatten@Table[CP[j, k], {k, $n, 1, -1}, {j, k, 1, -1}];
qc1 = QuantumCircuit[Sequence @@ SW[All], Sequence @@ gates]
```



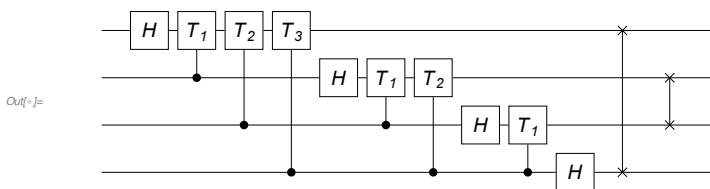
To verify the above quantum circuit model, we compare its matrix representation with the matrix for the discrete Fourier transform.

```
In[=]:= mat = Matrix[qc1] // TrigToExp;
new = FourierMatrix[Power[2, $n]];
new - mat // Simplify // MatrixForm
Out[=]/MatrixForm=
```

$$\begin{pmatrix} 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \end{pmatrix}$$

One can also reverse the order of qubits later to get the following quantum circuit model for the quantum Fourier transform.

```
In[=]:= gates = Flatten@Table[CP[j, k], {k, 1, $n}, {j, k, $n}];
qc2 = QuantumCircuit[Sequence @@ gates, Sequence @@ SW[All]]
```



Again, verify it by comparing the matrix representation with the matrix for the discrete Fourier transform.

```
In[=]:= mat = Matrix[qc2] // TrigToExp;
new = FourierMatrix[Power[2, $n]];
new - mat // Simplify // MatrixForm
Out[=]/MatrixForm=
```

$$\begin{pmatrix} 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \end{pmatrix}$$

4.3.3 Semiclassical Implementation

The semiclassical implementations is not only just interesting from a physics point of view but also extremely interesting with the current level of technology (Griffiths & Niu, 1996). As it does not require any entanglement, the most fragile quantum information resource and hence difficult to maintain, many experimenters are using this implementation whenever the QFT and/or the quantum phase estimation (see Section 4.4) is in needs (Chiaverini, 2005; Higgins *et al.*, 2007).

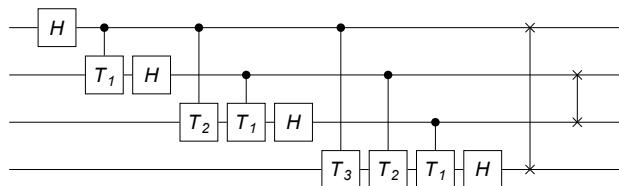
The line of arguments in the original work (Griffiths & Niu, 1996) to derive the semiclassical implementation of the quantum Fourier transformation is interesting in its own right, and offers another physically-motivated derivation of the quantum Fourier transform algorithm. Here, however, we will not repeat the original arguments. Instead, we make use the relation between the quantum Fourier transform and its inverse. The inverse quantum Fourier transform is given by (Problem 4.4)

$$\hat{U}_{\text{QFT}}^\dagger |X_y\rangle = \frac{1}{2^{n/2}} \sum_x |X_x\rangle e^{-ixp_y}, \quad (4.61)$$

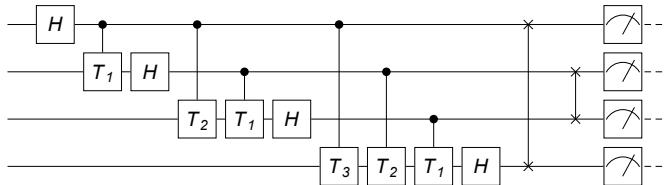
which follows directly from the orthgonality relation

$$\frac{1}{2^n} \sum_y e^{i(x-x')p_y} = \delta_{xx'} . \quad (4.62)$$

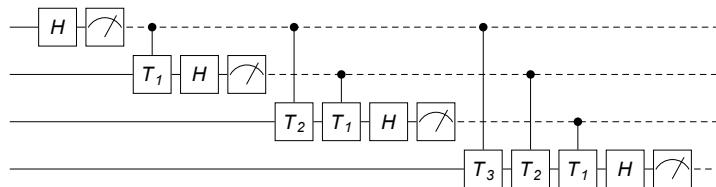
The expression (4.61) suggests that the inverse quantum Fourier transform is just another form of quantum Fourier transform—this is indeed a characteristic property common to any kind of Fourier transform. It implies that one can recover the origianl quantum Fourier transform from the inverse quantum Fourier transform, and vice versa, in several ways. For example, note that $\hat{U}_{\text{QFT}}^\dagger |X_y\rangle$ is an element of the conjugate basis because $e^{-ixp_y} = e^{ix(2\pi-p_y)}$ for any x and y . In fact, $\hat{U}_{\text{QFT}}^\dagger |X_y\rangle = |P_{2^n-y}\rangle$. More useful in the context of the semiclassical implementation of the quantum Fourier transform is to note that the inverse quantum Fourier transform $\hat{U}_{\text{QFT}}^\dagger$ is nothign but the quantum Fourier transform with the phase factor e^{ixp_y} replaced by e^{-ixp_y} . It corresonds to replacing the relative phase shifts $\hat{T}(\phi)$ in (4.59) by $T^\dagger(\phi)$. Conversely, if we start with the quantum circuit model in (4.59), take its Hermitian conjuate (i.e., inverse), and replace $\hat{T}^\dagger(\phi) \equiv \hat{T}(-\phi)$ with $\hat{T}(\phi)$, then we should recover the original quantum Fourier transform. Through this procedure, we get the following quantum circuit model



for the quantum Fourier transform. Compared with the quantum circuit model in (4.59), the above quantum circuit model has the Hadamard gate on a fixed qubit coming before the qubit “controls” the operation of the relative phase shift $\hat{T}(\phi)$ on other qubits. This difference brings a significant consequence. To see it, we put the measurements on the qubits explicitly,



According to the *principle of deferred measurement*, the measurement can be performed before the controlled unitary gates



This form is exactly the circuit model derived in [Griffiths & Niu \(1996\)](#). The important point is that after the measurement, the controlled unitary gates become just classical feedback controls, and require no entanglement.

Here we simulate the semiclassical implementation of the quantum Fourier transform. Again, we consider a quantum register of four qubits.

```
$n = 4;
SS = S[Range[$n], None];
```

Here defined are the controlled-phase gates.

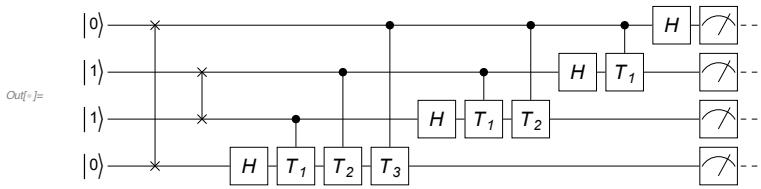
```
CP[j_, j_] := S[j, 6]
CP[j_, k_] := ControlledU[S[j],
  Phase[Pi Power[2, k - j], S[k], "Label" → Subscript["T", j - k]]
] /; j > k
CP[j_, k_] := ControlledU[S[j],
  Phase[Pi Power[2, j - k], S[k], "Label" → Subscript["T", k - j]]
] /; j < k

In[=] in = Ket @ Thread[SS → RandomChoice[{0, 1}, $n]];
LogicalForm[in, SS]
```

Out[=] $|0_{S_1}1_{S_2}1_{S_3}0_{S_4}\rangle$

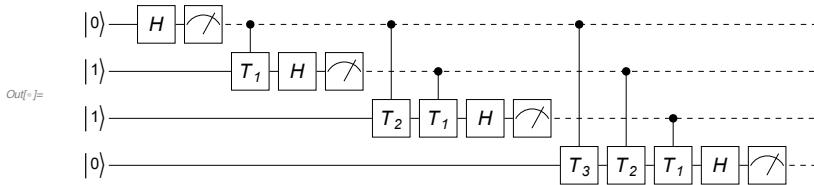
This is a quantum circuit model for the quantum Fourier transformation.

```
In[7]:= gates = Flatten@Table[CP[j, k], {k, $n, 1, -1}, {j, k, 1, -1}];
swaps = Table[SWAP[S[k], S[$n - k + 1]], {k, 1, $n / 2}];
qc1 = QuantumCircuit[LogicalForm[in, SS],
Sequence @@ swaps, Sequence @@ gates, Measurement[SS]]
```



This is the semiclassical implementation of the quantum Fourier transform.

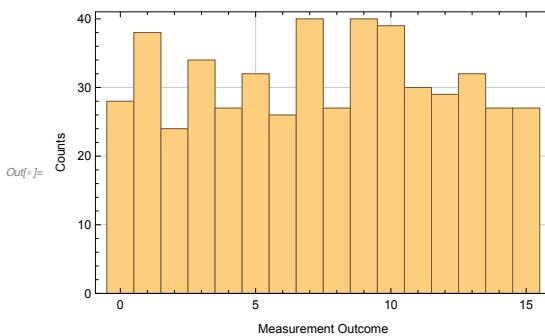
```
In[8]:= gates = Table[CP[j, k], {k, 1, $n}, {j, 1, k}];
gates = Flatten@MapIndexed[Append[#1, Measurement@S[#2]] &, gates];
swaps = Table[SWAP[S[k], S[$n - k + 1]], {k, 1, $n / 2}];
qc2 = QuantumCircuit[LogicalForm[in, SS], Sequence @@ gates]
```



To verify the equivalence of the two quantum circuit models, we compare the probability distributions.

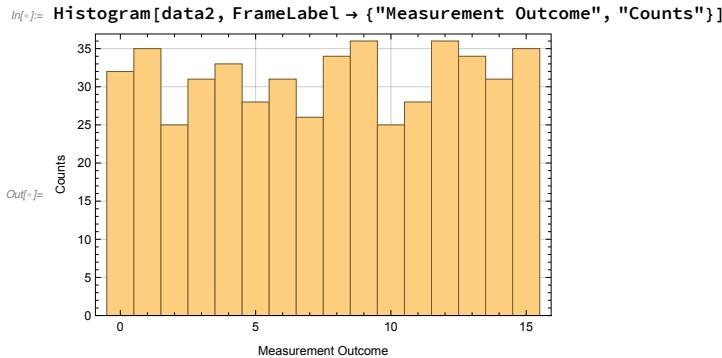
```
In[9]:= Timing[data1 = Table[out = ExpressionFor[qc1];
FromDigits[Readout[out, SS], 2], {500}];]
Out[9]= {63.5074, Null}
```

```
In[10]:= Histogram[data1, FrameLabel -> {"Measurement Outcome", "Counts"}]
```



Note that the bit strings of the measurement outcomes from the semiclassical model must be reversed .

```
In[11]:= Timing[data2 = Table[out = ExpressionFor[qc2];
FromDigits[Reverse@Readout[out, SS], 2], {500}];]
Out[11]= {18.8853, Null}
```



The discrepancy between the two distributions is due to the finite sample size.

Another way to test the equivalence of the fully quantum and semiclassical implementation of the quantum Fourier transformation is to take as the input state one that ends up in a product state.

```
In[6]:= vecs = FourierMatrix[Power[2, $n], FourierParameters → {0, -1}].Basis[SS];
in = RandomChoice[vecs];
in // LogicalForm
```

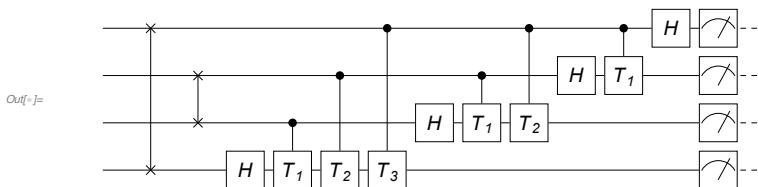
$$\frac{1}{4} \left| 0_{S_1} 0_{S_2} 0_{S_3} 0_{S_4} \right\rangle + \frac{1}{4} e^{\frac{3 i \pi}{8}} \left| 0_{S_1} 0_{S_2} 0_{S_3} 1_{S_4} \right\rangle + \frac{1}{4} e^{\frac{3 i \pi}{4}} \left| 0_{S_1} 0_{S_2} 1_{S_3} 0_{S_4} \right\rangle + \frac{1}{4} e^{-\frac{7 i \pi}{8}} \left| 0_{S_1} 0_{S_2} 1_{S_3} 1_{S_4} \right\rangle -$$

$$\frac{1}{4} i \left| 0_{S_1} 1_{S_2} 0_{S_3} 0_{S_4} \right\rangle + \frac{1}{4} e^{-\frac{i \pi}{8}} \left| 0_{S_1} 1_{S_2} 0_{S_3} 1_{S_4} \right\rangle + \frac{1}{4} e^{\frac{i \pi}{4}} \left| 0_{S_1} 1_{S_2} 1_{S_3} 0_{S_4} \right\rangle + \frac{1}{4} e^{\frac{5 i \pi}{8}} \left| 0_{S_1} 1_{S_2} 1_{S_3} 1_{S_4} \right\rangle -$$

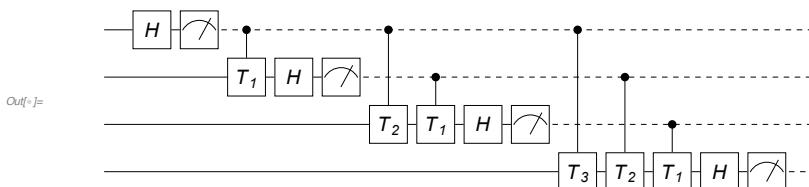
$$\frac{1}{4} \left| 1_{S_1} 0_{S_2} 0_{S_3} 0_{S_4} \right\rangle + \frac{1}{4} e^{-\frac{5 i \pi}{8}} \left| 1_{S_1} 0_{S_2} 0_{S_3} 1_{S_4} \right\rangle + \frac{1}{4} e^{-\frac{i \pi}{4}} \left| 1_{S_1} 0_{S_2} 1_{S_3} 0_{S_4} \right\rangle + \frac{1}{4} e^{\frac{i \pi}{8}} \left| 1_{S_1} 0_{S_2} 1_{S_3} 1_{S_4} \right\rangle +$$

$$\frac{1}{4} i \left| 1_{S_1} 1_{S_2} 0_{S_3} 0_{S_4} \right\rangle + \frac{1}{4} e^{\frac{7 i \pi}{8}} \left| 1_{S_1} 1_{S_2} 0_{S_3} 1_{S_4} \right\rangle + \frac{1}{4} e^{-\frac{3 i \pi}{4}} \left| 1_{S_1} 1_{S_2} 1_{S_3} 0_{S_4} \right\rangle + \frac{1}{4} e^{-\frac{3 i \pi}{8}} \left| 1_{S_1} 1_{S_2} 1_{S_3} 1_{S_4} \right\rangle$$

```
In[7]:= gates = Flatten@Table[CP[j, k], {k, $n, 1, -1}, {j, k, 1, -1}];
swaps = Table[SWAP[S[k], S[$n - k + 1]], {k, 1, $n/2}];
qc1 = QuantumCircuit[{in, "Label" → None},
Sequence @@ swaps, Sequence @@ gates, Measurement[SS]]
```



```
In[8]:= gates = Table[CP[j, k], {k, 1, $n}, {j, 1, k}];
gates = Flatten@MapIndexed[Append[#1, Measurement@S[#2]] &, gates];
swaps = Table[SWAP[S[k], S[$n - k + 1]], {k, 1, $n/2}];
qc2 = QuantumCircuit[{in, "Label" → None}, Sequence @@ gates]
```



```
In[7]:= out1 = ExpressionFor[qc1];
LogicalForm[out1, SS]
Out[7]= |1S11S20S31S4⟩
```

Again, recall that the order of qubits are reversed in the semiclassical implementation.

```
In[8]:= out2 = ExpressionFor[N@qc2] // Chop;
LogicalForm[out2, SS]
Out[8]= 1. |1S10S21S31S4⟩
```

From the quantum circuit models discussed above, we see that quantum Fourier transform requires an order of n^2 elementary gates. For example, consider the quantum circuit model in (4.59). There are n Hadamard gates, one for each qubit. The number controlled-unitary gates is given by $\sum_{k=1}^n (k - 1) = n(n - 1)/2$. There are $n/2$ SWAP gates as well, each of which requires three CNOT gates. Overall, we need $n(n + 4)/2 \sim \mathcal{O}(n^2)$ gates. This computational load is compared to $\mathcal{O}(n2^n)$ gates for the fast Fourier transform (FFT), the best classical algorithm for the discrete Fourier transform.

4.4 Quantum Phase Estimation (QPE)

Quantum phase estimation is a procedure to estimate the phase of the unknown eigenvalue of an eigenstate of a unitary operator. Like the quantum Fourier transformation, the quantum algorithm for quantum phase estimation is one of the key elements of many quantum algorithms including quantum factorization algorithm. In fact, the quantum Fourier transform is a part of the quantum phase estimation algorithm, and almost always appears in that way rather than independently. In this sense, quantum phase estimation is the key subroutine in most quantum algorithms. Indeed, all quantum algorithms known so far can be regarded as quantum phase estimation in one form or another (Cleve *et al.*, 1998).^{4.6}

4.4.1 Definition

Let \hat{U} be a unitary operator on the Hilbert space associated with a register consisting of n qubits. Suppose that the quantum register is known to be prepared in an eigenstate state $|\phi\rangle$ of \hat{U} . Its eigenvalue $e^{-2\pi i\phi}$ is desired but *unknown*. The quantum phase estimation (QPE) procedure is to get the phase value $2\pi\phi$ of the eigenvalue.

Note that the quantum phase estimation is directly related to the measurement. Recall that finding the eigenvalue $e^{-2\pi i\phi}$ reveals the corresponding eigenstate $|\phi\rangle$. As an unitary operator can be written as $\hat{U} = e^{-2\pi i\hat{Q}}$ for some observable \hat{Q} , the estimation of phase corresponds to the measurement of the quantity \hat{Q} . In this

^{4.6}Mathematically speaking, they belong to the class called the hidden subgroup problem.

sense, one can regard the quantum phase estimation procedure as a realization of quantum measurement on quantum computers. We will discuss this aspect in more detail in Section 4.4.4.

To get a general idea behind the quantum phase estimation procedure, take an auxiliary register of m qubits. We call it the “probe” register for the reason to be clear later in this section. We prepare the probe register in the superposition $\propto \sum_x |x\rangle$ of all elements in the logical basis. The superposition state can be generated by applying the Hadamard gate—see Eq. (2.18). The state vector of the total system becomes

$$\left(\hat{H}^{\otimes m}|0\rangle\right) \otimes |\phi\rangle = \frac{1}{2^{m/2}} \sum_{x=0}^{2^m-1} |x\rangle \otimes |\phi\rangle \quad (4.63)$$

Next, perform a unitary transformation on the total system defined by

$$\hat{U}_{\text{QPE}} : |x\rangle \otimes |y\rangle \mapsto |x\rangle \otimes \left(\hat{U}^x |y\rangle\right), \quad (4.64)$$

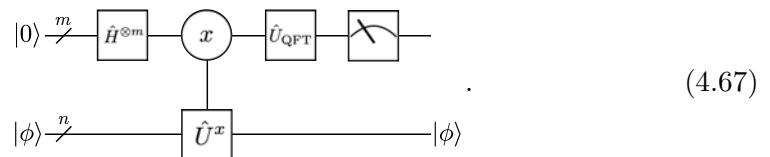
where $|x\rangle$ ($x = 0, 1, \dots, 2^m - 1$) belongs to the probe register and $|y\rangle$ ($y = 0, 1, \dots, 2^n - 1$) to the native register. The transformation is a controlled unitary gate, performing the transformation \hat{U} repeatedly depending on the value of y on the native register. Upon the controlled unitary transformation, the state vector becomes

$$\begin{aligned} \frac{1}{2^{m/2}} \sum_x |x\rangle \otimes \left(\hat{U}^x |\phi\rangle\right) &= \frac{1}{2^{m/2}} \sum_x |x\rangle \otimes \left(|\phi\rangle e^{-2\pi i x \phi}\right) \\ &= \left(\frac{1}{2^{m/2}} \sum_{x=0}^{2^m-1} |x\rangle e^{-2\pi i x \phi}\right) \otimes |\phi\rangle \end{aligned} \quad (4.65)$$

Note that the original register still remains in the same state $|\phi\rangle$, whereas the probe register has picked up the relative phase shift proportional to x , $e^{-2\pi i x \phi}$. The state in (4.65) stored on the probe register is thus formally the same as the (inverse) quantum Fourier transform—see (4.38) and (4.61). For the moment, let us assume that ϕ ($0 \leq \phi < 1$) takes one of the discrete values

$$\frac{0}{2^m}, \frac{1}{2^m}, \frac{2}{2^m}, \dots, \frac{2^m - 1}{2^m}. \quad (4.66)$$

In this case, performing the quantum Fourier transform \hat{U}_{QFT} on the probe register brings it to the logical basis state $|2^m \phi\rangle$. A subsequent measurement on the probe register in the logical basis reveals the value ϕ . The procedure described above is summarized in the following diagram



The normalized phase ϕ is a continuous parameter. In general, it takes values other than the discrete values listed in (4.66). In such a case, the number m of qubits in the probe register is not sufficient to accurately estimate the phase ϕ . The measurement randomly produces different values. Fortunately, the probability for a particular value distributed sharply—the larger the probe register is, the sharper the probability distribution becomes—around the value $2^m\phi$. We will come back to this point below.

4.4.2 Implementation

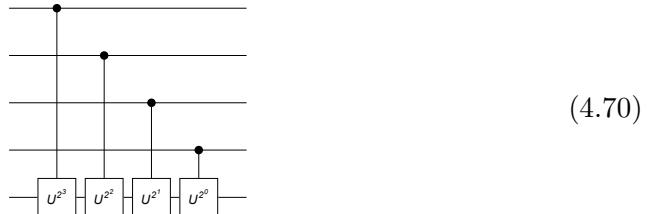
Let us now discuss the actual implementation of the individual steps. The preparation of the probe register in the overall superposition state is achieved by using the Hadamard gate (see Section 2.1.2)

$$|0\rangle^{\otimes m} \rightarrow \hat{H}^{\otimes m} |0\rangle^{\otimes m} = \frac{1}{2^{m/2}} \sum_{x=0}^{2^m-1} |x\rangle. \quad (4.68)$$

Explicitly writing, the controlled- \hat{U}^x transformation in Eq. (4.64) reads as

$$|x\rangle \otimes \hat{U}^x |y\rangle = |x_1\rangle \otimes |x_2\rangle \otimes \cdots \otimes |x_m\rangle \otimes \hat{U}^{x_1 2^{m-1}} \hat{U}^{x_2 2^{m-2}} \cdots \hat{U}^{x_m 2^0} |y\rangle. \quad (4.69)$$

It is a product of controlled- \hat{U}^{2^j} gates depending on the value x_j of the j th qubit in the probe register. It is thus implemented by the following quantum circuit model



Here, it is noted that an efficient quantum phase estimation algorithm requires an efficient implementation of the controlled- U^{2^j} gates. One of the most common unitary transformations that allows for efficient implementation is the *modular multiplication*, which is used in the order-finding problem to be discussed in Section 4.5.2.

The controlled- \hat{U}^x gate transforms the state of the total system as

$$\frac{1}{2^{m/2}} \sum_{x=0}^{2^m-1} |x\rangle \otimes |\phi\rangle \rightarrow \left(\frac{1}{2^{m/2}} \sum_{x=0}^{2^m-1} |x\rangle e^{-2\pi i x \phi} \right) \otimes |\phi\rangle \quad (4.71)$$

The final quantum Fourier transformation on the probe register transforms the state as

$$\frac{1}{2^{m/2}} \sum_x |x\rangle e^{-2\pi i x \phi} \rightarrow \sum_y |y\rangle \left(\frac{1}{2^m} \sum_x |y\rangle e^{2\pi i (y - 2^m \phi) x / 2^m} \right) \quad (4.72)$$

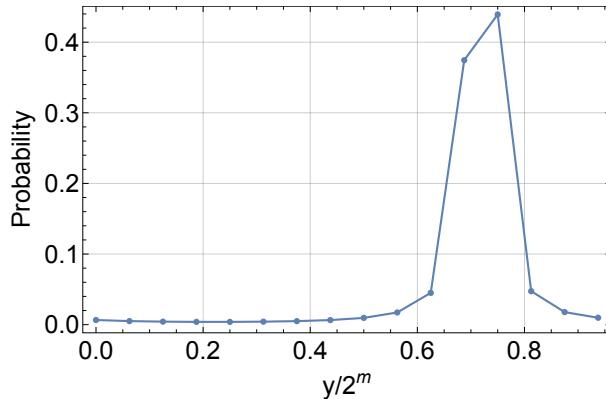


Figure 4.2: The probability P_y to estimate the phase ϕ as $y/2^m$ by using $m = 4$ “probe” qubits when the true phase value is $\phi = 18/25$.

When $2^m\phi$ is one of the discrete values $0, 1, 2, \dots, 2^m - 1$, the quantum Fourier transform brings the probe to the single logical basis state $|2^m\phi\rangle$. It follows from the authogonality relation

$$\frac{1}{2^m} \sum_{k=0}^{2^m-1} e^{2\pi i(x-y)k/2^m} = \delta_{xy}. \quad (4.73)$$

In general, it is not the case. The state is a superposition of different logical states. Upon the measurement on the probe register in the logical basis, the probability to get the outcome y is given by

$$P_y = \left| \frac{1}{2^m} \sum_{x=0}^{2^m-1} e^{i2\pi(y-2^m\phi)x/2^m} \right|^2 = \frac{1}{2^{2m}} \left| \frac{1 - e^{2\pi i(y-2^m\phi)}}{1 - e^{2\pi i(y-p)/2^m}} \right|^2. \quad (4.74)$$

For reasonably large m , the probability P_y is sharply peaked around $2^m\phi$ (see Fig. 4.2). It is precisely 1 only when $2^m\phi$ is one of the integers $0, 1, \dots, 2^m - 1$ as mentioned above.

We take an ancillary register consisting of four qubits, which we call the “probe” register.

```
$m = 4; (* the number of qubits in the "probe" register *)
prb = Range[$m]; (* the "probe" register *)
sys = $m + 1; (* the "system" register *)
```

We consider a rotation around the z-axis for the unitary operator. In this case, the logical basis consists of the eigenstates of the unitary operator.

```
ϕ = 18 / 25;
```

Here is the controlled – $U^{2^{m-j}}$ operator to be used in the phase estimation algorithm.

```

ctrlU[j_] := ControlledU[S[j], Phase[-2 Pi φ * Power[2, $m - j], S[sys]],
  "Label" → Superscript["U", Superscript["2", ToString[$m - j]]],
  "LabelSize" → 0.65]

In[7]:= qc = QuantumCircuit[LogicalForm[Ket[], S@prb],
  Ket[S[sys] → 1], {S[prb, 6], "LabelSize" → 0.8}, ctrlU[1],
  ctrlU[2], ctrlU[3], ctrlU[4], QuantumFourierTransform[S@prb]]

```

Out[7]=

In this particular example, we already know that the “system” register remains in its initial state and what the initial state is. To make the process general, however, we take the partial trace over the system register to focus on the “probe” register.

```

out = Matrix[N@qc];
rho = PartialTrace[out, {sys}];

```

From the density matrix of the probe register, we can get the probability to read out the possible values of the normalized phase.

```

In[8]:= probability = Transpose@{Range[0, 2^$m - 1] / 2^$m, Normal@Chop@Diagonal@rho};
g = ListLinePlot[probability,
  PlotRange → {{0, 1}, All}, PlotMarkers → Automatic,
  FrameLabel → {"φ", "Probability"}]

```

Out[8]=

φ	Probability
0.00	0.00
0.05	0.00
0.10	0.00
0.15	0.00
0.20	0.00
0.25	0.00
0.30	0.00
0.35	0.00
0.40	0.00
0.45	0.00
0.50	0.00
0.55	0.02
0.60	0.05
0.65	0.05
0.70	0.38
0.75	0.45
0.80	0.05
0.85	0.02
0.90	0.01
0.95	0.01
1.00	0.00

4.4.3 Accuracy

We now estimate the error bound of the phase estimation.^{4.7} We split $2^m\phi$ into the integer part p and the fractional part δ so that $2^m\phi = p + \delta$ with $0 \leq \delta < 1$. The fractional part δ cannot be stored in the m -qubit probe register and causes an error in the estimated phase. For a given resource of finite size, we cannot avoid it.

One can get the best estimate when the measurement outcome y points correctly to p or $p + 1$. The error $\varepsilon := |y/2^m - \phi|$ in the estimated phase would be less than

^{4.7}A different method and probability bound are presented in Nielsen & Chuang (2011).

2^{-m} . The probability P_y is distributed sharply (for large m) around p and $p+1$. The probabilities at $y = p$ and $y = p+1$ are bigger than at any other values of y . Therefore, it is useful to bound the probability for the error in the estimated phase is less than 2^{-m} ,

$$P(\varepsilon < 2^{-m}) = P_p + P_{p+1}. \quad (4.75)$$

Unfortunately, the probability distribution P_y has a finite width, and the outcome y does not always result in the desired value. Nevertheless, P_y at $y = p$ and $y = p+1$ are bigger than at any other values of y .

It follows from (4.74) that

$$P(\varepsilon < 2^{-m}) = \frac{1}{2^{2m}} \left| \frac{\sin(\pi\delta)}{\sin(\pi\delta/2^m)} \right|^2 + \frac{1}{2^{2m}} \left| \frac{\sin(\pi(1-\delta))}{\sin(\pi(1-\delta)/2^m)} \right|^2. \quad (4.76)$$

As a function of δ , it is symmetric around $\delta = 1/2$, that is, $1/2 - \delta - > \delta$ does not change the expression. Further, it has maximum 1 for $\delta = 0$ because in this case $P_p = 1$ and no error can arise. It has minimum for $\delta = 1/2$ because in this case $P_p = P_{p+1}$. We thus have the inequality

$$P(\varepsilon < 2^{-m}) \geq \frac{1}{2^{2m-1}} \left| \frac{\sin(\pi/2)}{\sin(\pi/2^{m+1})} \right|^2 = \frac{1}{2^{2m-1} \sin^2(\pi/2^{m+1})}. \quad (4.77)$$

Note that $(\pi/2^{m+1})/\sin(\pi/2^{m+1})$ monotonically decreases and converges to 1 as m increases to the infinity. Therefore, we have

$$P(\varepsilon < 2^{-m}) \geq \frac{8}{\pi^2} \approx 0.81. \quad (4.78)$$

As m increases, we can improve the accuracy with error less than 2^{-m} with success probability larger than 0.81.

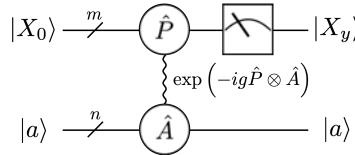
4.4.4 Simulation of von Neumann Measurement

The von Neumann scheme of measurement (see Section 1.3.1) is an idealistic mechanism to implement a projection measurement. As already mentioned, the quantum phase estimation procedure is in spirit equivalent to the von Neumann scheme of measurement. In fact, quantum phase estimation was put forward to simulate the von Neumann scheme on quantum computers (Kitaev, 1996, 1997). The equivalence of the von Neumann scheme of measurement and the quantum phase estimation procedure have also inspired the method of direct tomography of wave functions and its variants (Lundeen *et al.*, 2011; Vallone & Dequal, 2016). Therefore, it will be interesting for the scientific as well as heuristic purposes to examine the equivalence in more detail.

We consider a “system” consisting of n qubits, and pick a “probe” of m qubits. To maintain the physical nature of the von Neumann scheme, for the probe let us

consider two bases, $\{|X_x\rangle\}$ and $\{|P_x\rangle\}$ ($x = 0, 1, 2, \dots, 2^m - 1$), that are conjugate to each other and related by the quantum Fourier transform $|P_x\rangle = \hat{U}_{\text{QFT}}|X_x\rangle$. We call them the “position” and “momentum” basis, respectively (see Section 4.3.1). To make the argument simpler, we assume the system is in a definite yet unknown eigenstate $|a\rangle$ of \hat{A} . We want to find the eigenvalue a by a measurement procedure.

Following the von Neumann scheme, we prepare the probe in the state $|X_0\rangle$, which refers to a definite position. We let the system and probe interact with each other by coupling the “momentum” operator \hat{P} of the probe to the observable \hat{A} of the system with strength g . After the interaction for a finite duration of time, the probe is measured in the basis $|X_y\rangle$. This is illustrated in the following schematic diagram



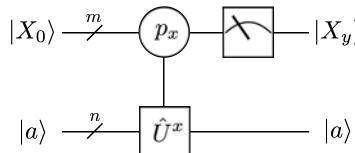
We rewrite the interaction unitary operator as [to be compared with (1.46)]

$$\hat{U}_{\text{int}} = \exp\left(-ig\hat{P} \otimes \hat{A}\right) = \sum_x |P_x\rangle\langle P_x| \otimes e^{-igp_x\hat{A}} = \sum_x |P_x\rangle\langle P_x| \otimes \hat{U}^x \quad (4.79)$$

where the operator \hat{U} acting on the system has been defined by

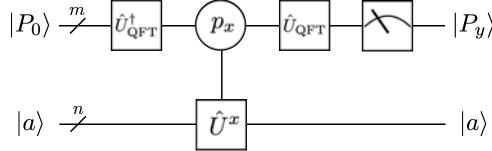
$$\hat{U} = \exp\left[-i\left(\frac{2\pi g}{2^n}\right)\hat{A}\right]. \quad (4.80)$$

In this new interpretation, the operator \hat{U} is repeatedly operated x times on the system depending on the wave number p_x (eigenvalue of \hat{P}) in the probe. This interpretation is seemingly opposite to the original idea of the von Neumann scheme, where the translation operator \hat{T}_a is operated on the probe depending on the eigenvalue a of \hat{A} in the system. This reinterpretation is depicted schematically in the following diagram [to be compared with the diagram in (1.47)]

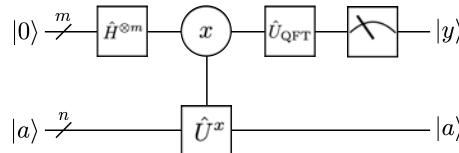


Note that the probe register controls the operation of \hat{U} on the system in the “momentum” basis $\{|P_y\rangle\}$, and hence the momentum basis is the computational basis (logical basis). On the other hand, the von Neumann scheme requires the measurement to be performed in the “position” basis $\{|X_y\rangle\}$. As discussed in Section 2.5, any measurement in a basis different from the logical basis can be achieved by applying a proper unitary transformation corresponding to the basis

change. In this case, it is the quantum Fourier transform. Similarly, one can obtain the input state $|X_0\rangle$ by acting $\hat{U}_{\text{QFT}}^\dagger$ on $|P_0\rangle$. These changes lead to the schematic diagram



Finally, with all bases changed to the logical basis $\{|P_x\rangle\}$, we can drop the addition labels ‘ P ’ and ‘ p ’ from the states and eigenvalues. Recall also that the action of the quantum Fourier transform on $|P_0\rangle$ can be replaced by the Hadamard gates, $\hat{U}_{\text{QFT}}^\dagger |P_0\rangle = \hat{H}^{\otimes m} |P_0\rangle$ (see Problem 4.5). Therefore, the von Neumann measurement can be depicted as the following diagram, which is identical to the diagram (4.67) corresponding to the quantum phase estimation procedure.



4.5 Applications

Before concluding the chapter, let us examine two example problems where the quantum Fourier transform and the quantum phase estimation can be applied. Fourier transform is particularly useful for periodic effects. It is thus natural to use the quantum Fourier transform to find the unknown period of a given function. The order-finding problem is a specific example, where the function is the *modular exponentiation*. As mentioned earlier, however, the quantum Fourier transform cannot be used independently although it is the key part. One needs a procedure to induce the relative phase shifts, which can be extracted with the quantum Fourier transform. The procedure is a type of quantum phase estimation. Mathematically, the period-finding and order-finding problem belong to a wider class of problems known as the *hidden subgroup problem*.

4.5.1 The Period-Finding Algorithm

Let $f : \{0, 1\}^m \rightarrow \{0, 1\}^n$ be a periodic function, and

$$f(x + \lambda) = f(x) \quad (4.81)$$

for all x and fixed λ . The period-finding problem is to find the unknown (primary) period λ with the least queries to the function f . For simplicity, we assume that λ

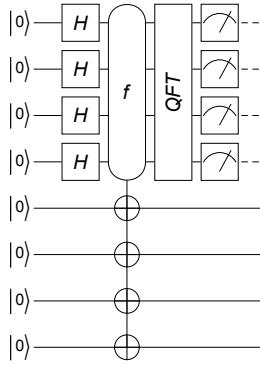


Figure 4.3: A quantum circuit model for the period-finding algorithm.

is a divider of 2^m . Here we also assume that the quantum oracle \hat{U}_f corresponding to the function f is efficiently implemented for the mapping

$$\hat{U}_f : |x\rangle \otimes |y\rangle \mapsto |x\rangle \otimes |y \oplus f(x)\rangle. \quad (4.82)$$

Given the sequence of function values $f(0), f(1), \dots, f(\lambda - 1)$, one can perform the discrete Fourier transform of the sequence to get

$$F_k := \frac{1}{\sqrt{\lambda}} \sum_{x=0}^{\lambda-1} f(x) e^{-2\pi i k x / \lambda}. \quad (4.83)$$

Analogously, it is also possible to define new states^{4.8}

$$|F_k\rangle = \frac{1}{\sqrt{\lambda}} \sum_{x=0}^{\lambda-1} |f(x)\rangle e^{+2\pi i k x / \lambda} \quad (4.84)$$

for $k = 0, 1, 2, \dots, \lambda - 1$, using the discrete Fourier transform of the sequence of states $|f(x)\rangle$ with $x = 0, 1, 2, \dots, \lambda - 1$. It should be noted that in general, the new states $|F_k\rangle$ may not be linearly independent with each other because $|f(x)\rangle = |f(y)\rangle$ for some $0 \leq x, y < \lambda$.^{4.9}

The period-finding algorithm is summarized in the quantum circuit model in Fig. 4.3. We prepare the control register in the state $2^{-m/2} \sum_x |x\rangle$. The quantum oracle (4.82) then makes the transformation

$$\frac{1}{2^{m/2}} \sum_{x=0}^{2^m-1} |x\rangle \otimes |0\rangle \rightarrow \frac{1}{2^{m/2}} \sum_{x=0}^{2^m-1} |x\rangle \otimes |f(x)\rangle. \quad (4.85)$$

^{4.8}Here we take the opposite sign in the exponent of the phase factor $e^{2\pi i k x / \lambda}$ to be consistent with the quantum Fourier transform in (4.38).

^{4.9}In this respect, the period-finding problem in the presented form is slightly different from the hidden subgroup problem. In the hidden subgroup problem, the function $f : \mathcal{G} \rightarrow \mathcal{S}$ from a group \mathcal{G} to a set \mathcal{S} is assumed to separate the cosets. In other words, $f(x) = f(y)$ if and only if x and y belong to the same coset of the given subgroup $\mathcal{H} \subset \mathcal{G}$.

We now replace the states $|f(x)\rangle$ with $|F_k\rangle$ in (4.84) using the inverse relation

$$|f(x)\rangle = \frac{1}{\sqrt{\lambda}} \sum_{k=0}^{\lambda-1} |F_k\rangle e^{-2\pi i k x / \lambda}, \quad (4.86)$$

which leads to the expression for the state of the total system

$$\frac{1}{\sqrt{\lambda}} \sum_{k=0}^{\lambda-1} \left(\frac{1}{2^{m/2}} \sum_{x=0}^{2^m-1} |x\rangle e^{-2\pi i x k / \lambda} \right) \otimes |F_k\rangle. \quad (4.87)$$

It reveals that the quantum oracle has induced the relative phase shifts in the control register that are proportional to the logical value x . As in the quantum phase estimation procedure discussed previously in Section 4.4, one can extract the relative phase shifts by employing the quantum Fourier transform—see Eqs. (4.71) and (4.72). It corresponds to estimating the (normalized) “phase” k/λ , from which one can find the period λ (see below). In this sense, one can regard that the period-finding problem falls in the category of quantum phase estimation.

Of course, one apparent difference is that the relative phase shifts are induced by the quantum oracle in this case. Further, unlike the standard quantum phase estimation procedure, the phase k/λ to be estimated is randomly selected from the values corresponding to $k = 0, 1, \dots, \lambda - 1$. It is possible that the observed k may have a common divider with λ so that $k/\lambda = k'/\lambda'$ for another pair of integers k' and λ' . For this reason, one cannot determine the period λ completely from the recorded value k/λ . The denominator (and integer multiples of it) is only a candidate for the period λ . However, it does not pose a serious obstacle as it is easy to check if a candidate is a true period or not by evaluating the function for a few input values. In passing, note that the probability P_k for k to take a particular value,

$$P_k = \frac{\langle F_k | F_k \rangle}{\lambda}, \quad (4.88)$$

is not uniformly distributed—it varies with k .^{4.10} Again, this is because it is possible $f(x) = f(y)$ for some $0 \leq x, y < \lambda$.

We demonstrate the basic ideas in the period-finding algorithm. More complete example follows below.

```
$n = 4;
$N = Power[2, $n];
```

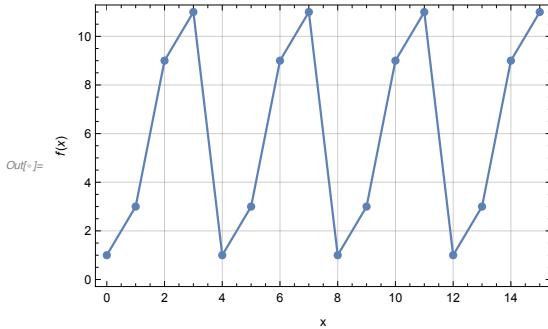
We take an example of the modular exponential function, $f(x) = a^x \pmod{L}$ for a fixed a and L . Here a and L are assumed to be coprimes.

```
$a = 3;
f[k_] := PowerMod[$a, k, $N]
```

^{4.10}This is another difference from the hidden subgroup problem.

The function is periodic.

```
In[7]:= ListPlot[Table[{k, f[k]}, {k, 0, $N - 1}], Joined → True,
  PlotMarkers → Automatic, FrameLabel → {"x", HoldForm@f[x]}]
```

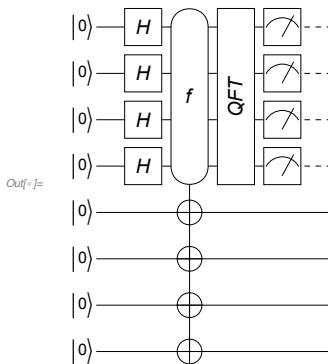


The classical oracle (in the binary form) corresponding to the function f (expressed in usual integers).

```
g[x __] := IntegerDigits[f[FromDigits[{x}, 2]], 2, $n]
```

Here is the quantum circuit model of the algorithm.

```
In[8]:= cc = Range[$n];
tt = Range[$n + 1, $n + $n];
qc0 = QuantumCircuit[
  LogicalForm[Ket[], Join[S@cc, S@tt]], S[cc, 6], Oracle[g, S@cc, S@tt],
  QuantumFourierTransform[S@cc]];
qc = QuantumCircuit[qc0, Measurement[S@cc]]
```



```
In[9]:= out = ExpressionFor[qc];
LogicalForm[out, Join[S@cc, S@tt]]
result = Readout[out, S@cc]
Out[9]=  $\frac{1}{2} |\theta_{S_1} 1_{S_2} \theta_{S_3} 0_{S_4} 0_{S_5} 0_{S_6} 0_{S_7} 1_{S_8}\rangle + \frac{1}{2} i |\theta_{S_1} 1_{S_2} \theta_{S_3} 0_{S_4} 0_{S_5} 0_{S_6} 1_{S_7} 1_{S_8}\rangle - \frac{1}{2} |\theta_{S_1} 1_{S_2} \theta_{S_3} 0_{S_4} 1_{S_5} 0_{S_6} 0_{S_7} 1_{S_8}\rangle - \frac{1}{2} i |\theta_{S_1} 1_{S_2} \theta_{S_3} 0_{S_4} 1_{S_5} 0_{S_6} 1_{S_7} 1_{S_8}\rangle$ 
```

```
Out[9]= {0, 1, 0, 0}
```

From the measurement outcome, one can find the period. In this case, the period is 4.

```
In[1]:= frac = FromDigits[result, 2] / $N
Out[1]=  $\frac{1}{4}$ 
```

For a more complete overview, let us examine all possible outcomes.

```
In[2]:= new = ExpressionFor[qc0];
LogicalForm[new, Join[S@cc, S@tt]]
```

$$\begin{aligned} \text{Out[2]}= & \frac{1}{4} \left| 0_{S_1} 0_{S_2} 0_{S_3} 0_{S_4} 0_{S_5} 0_{S_6} 0_{S_7} 1_{S_8} \right\rangle + \frac{1}{4} \left| 0_{S_1} 0_{S_2} 0_{S_3} 0_{S_4} 0_{S_5} 0_{S_6} 1_{S_7} 1_{S_8} \right\rangle + \\ & \frac{1}{4} \left| 0_{S_1} 0_{S_2} 0_{S_3} 0_{S_4} 1_{S_5} 0_{S_6} 0_{S_7} 1_{S_8} \right\rangle + \frac{1}{4} \left| 0_{S_1} 0_{S_2} 0_{S_3} 0_{S_4} 1_{S_5} 0_{S_6} 1_{S_7} 1_{S_8} \right\rangle + \frac{1}{4} \left| 0_{S_1} 1_{S_2} 0_{S_3} 0_{S_4} 0_{S_5} 0_{S_6} 0_{S_7} 1_{S_8} \right\rangle + \\ & \frac{1}{4} \left| 0_{S_1} 1_{S_2} 0_{S_3} 0_{S_4} 0_{S_5} 0_{S_6} 1_{S_7} 1_{S_8} \right\rangle - \frac{1}{4} \left| 0_{S_1} 1_{S_2} 0_{S_3} 0_{S_4} 1_{S_5} 0_{S_6} 0_{S_7} 1_{S_8} \right\rangle - \\ & \frac{1}{4} \left| 0_{S_1} 1_{S_2} 0_{S_3} 0_{S_4} 1_{S_5} 0_{S_6} 1_{S_7} 1_{S_8} \right\rangle + \frac{1}{4} \left| 1_{S_1} 0_{S_2} 0_{S_3} 0_{S_4} 0_{S_5} 0_{S_6} 0_{S_7} 1_{S_8} \right\rangle - \frac{1}{4} \left| 1_{S_1} 0_{S_2} 0_{S_3} 0_{S_4} 0_{S_5} 0_{S_6} 1_{S_7} 1_{S_8} \right\rangle + \\ & \frac{1}{4} \left| 1_{S_1} 0_{S_2} 0_{S_3} 0_{S_4} 1_{S_5} 0_{S_6} 0_{S_7} 1_{S_8} \right\rangle - \frac{1}{4} \left| 1_{S_1} 0_{S_2} 0_{S_3} 0_{S_4} 1_{S_5} 0_{S_6} 1_{S_7} 1_{S_8} \right\rangle + \frac{1}{4} \left| 1_{S_1} 1_{S_2} 0_{S_3} 0_{S_4} 0_{S_5} 0_{S_6} 0_{S_7} 1_{S_8} \right\rangle - \\ & \frac{1}{4} \left| 1_{S_1} 1_{S_2} 0_{S_3} 0_{S_4} 0_{S_5} 0_{S_6} 1_{S_7} 1_{S_8} \right\rangle - \frac{1}{4} \left| 1_{S_1} 1_{S_2} 0_{S_3} 0_{S_4} 1_{S_5} 0_{S_6} 0_{S_7} 1_{S_8} \right\rangle + \frac{1}{4} \left| 1_{S_1} 1_{S_2} 0_{S_3} 0_{S_4} 1_{S_5} 0_{S_6} 1_{S_7} 1_{S_8} \right\rangle \end{aligned}$$

```
In[3]:= data = Table[out = Measurement[new, S@cc];
FromDigits[Readout[out, S@cc], 2], {15}]
```

$$\text{Out[3]}= \{4, 4, 0, 0, 4, 8, 12, 12, 0, 12, 0, 4, 8, 12, 12\}$$

```
In[4]:= sec = Union[data / $N]
```

$$\text{Out[4]}= \left\{ 0, \frac{1}{4}, \frac{1}{2}, \frac{3}{4} \right\}$$

The period must be 4, which agrees with the true value.

A more serious hurdle is that the estimated phase ϕ is only approximate to the precise value k/λ , $\phi \approx k/\lambda$, because the register is of finite size. This difference makes it more complicated to determine the period λ from the estimated phase ϕ . Fortunately, continued fraction representation provides an efficient method to find the period.

A *continued fraction* is an expression of the form

$$a_0 + \frac{1}{a_1 + \frac{1}{a_2 + \frac{1}{\ddots + \frac{1}{a_n}}}}, \quad (4.89)$$

where a_0, a_1, \dots, a_n are positive integers.^{4.11} It is often denoted by

$$[a_0; a_1, a_2, \dots, a_n]. \quad (4.90)$$

One can represent any real number with a unique continued fraction.^{4.12} For a given number, the procedure to find a continued fraction is simple. For example,

^{4.11}There are several other conventions for continued fractions. Here we only use the so-called *simple* or *canonical* continued fractions.

^{4.12}For a finite continued fraction, there is a trivial ambiguity such as $[a_0; a_1, a_2, \dots, a_n] = [a_0; a_1, a_2, \dots, (a_n - 1), 1]$. We regard such continued fractions are equivalent.

consider $13/5$. We first split it into its integer part is 2 and the fractional part $3/5$, $13/5 = 2 + 3/5$. Next we take the reciprocal of $3/5$ and split it into the integer and fractional parts, $5/3 = 1 + 2/3$. By repeating the procedure, we find that

$$\frac{13}{5} = 2 + \frac{1}{1 + \frac{1}{1 + \frac{1}{2}}}, \quad (4.91)$$

or equivalently,

$$\frac{13}{5} = [2; 1, 1, 2]. \quad (4.92)$$

The well-known Euclidean algorithm, dating back to Euclid's book *Elements*, efficiently finds a continued fraction representing a number. A continued fraction may be terminated by a finite n . Then it is said to be finite. If a continued fraction continues the iteration without termination, it is called an infinite continued fraction. The continued fraction $[a_0; a_1, a_2, \dots, a_m]$ is called the m th convergent of $[a_0; a_1, a_2, \dots, a_m, a_{m+1}, \dots, a_n]$ ($0 \leq m \leq n$).

Not so common in daily lives, continued fraction representations are more natural mathematically, in many respects, than the decimal representations or other similar representations with a fixed base. To see it, consider the following properties: The continued fraction for any rational number is finite. It is in contrast with the decimal representations. For example, $5/3 = [1; 1, 2]$ whereas $5/3 = 1.666\dots$. On the other hand, continued fractions for irrational numbers are infinite. For example, the golden ratio is represented by the continued fraction, $(1 + \sqrt{5})/2 = [1; 1, 1, 1, \dots]$. Further, the continued fractions for a number and its reciprocal are identical except for a shift by one place. In other words, $[a_0; a_1, a_2, \dots, a_n]$ and $[0; a_0, a_1, a_2, \dots, a_n]$ are reciprocal of each other.

The method to determine the period λ from the estimated phase $\phi \approx k/\lambda$ is based on a theorem in number theory. It states^{4.13} that if

$$\left| \frac{k}{\lambda} - \phi \right| \leq \frac{1}{2\lambda^2} \quad (4.93)$$

for any integers k and λ , then k/λ is a convergent of the continued fraction for ϕ . Note that ϕ is accurate to m bits. The error is less than 2^{-m} with probability higher than 0.81 (Section 4.4.3). Since $\lambda \ll 2^m$, the condition in (4.93) is satisfied. Therefore, the continued fraction of the estimated phase ϕ gives coprimes k'/λ' such that $k'/\lambda' = k/\lambda$. λ' and integer multiples of λ' are candidates for the period. We then evaluate (classically) the function to check if λ (or any integer multiple of λ') is the true period λ of the function.

Now we consider a full period-finding problem. We consider a four-qubit system.

```
$n = 4;
$N = Power[2, $n];
```

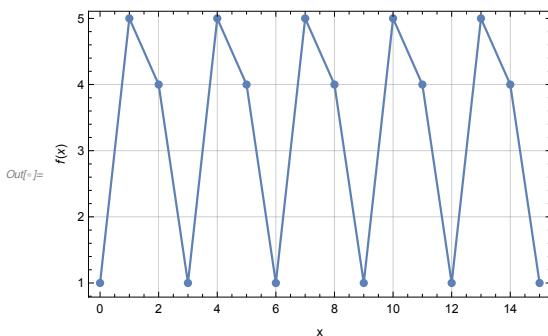
^{4.13}A proof is found in Nielsen & Chuang (2011, Appendix 4).

We define an arbitrary function with period 3.

```
f[0] = 1;
f[1] = 5;
f[2] = 4;
f[x_] := f[Mod[x, 3]]
```

This plot illustrates that the function is indeed periodic with period 3.

```
In[7]:= ListPlot[Table[{k, f[k]}, {k, 0, $N - 1}], Joined → True,
PlotMarkers → Automatic, FrameLabel → {"x", HoldForm@f[x]}]
```

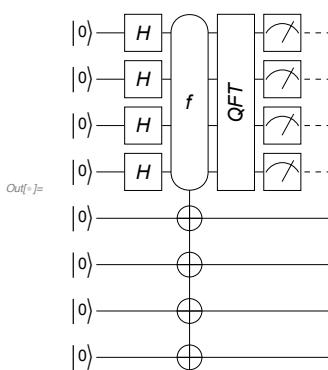


The classical oracle (in the binary form) corresponding to the function f (expressed in usual integers).

```
g[x__] := IntegerDigits[f[FromDigits[{x}, 2]], 2, $n]
```

Here is the quantum circuit model of the algorithm.

```
In[8]:= cc = Range[$n];
tt = Range[$n + 1, $n + $n];
qc0 = QuantumCircuit[
  LogicalForm[Ket[]], Join[S@cc, S@tt]], S[cc, 6], Oracle[g, S@cc, S@tt],
  QuantumFourierTransform[S@cc]];
qc = QuantumCircuit[qc0, Measurement[S@cc]]
```



This is a full run of the above quantum circuit model.

```
In[7]:= Timing[out = ExpressionFor[qc];]
LogicalForm[out, Join[S@cc, S@tt]];
result = Readout[out, S@cc]

Out[7]= {7.02321, Null}

Out[8]= {1, 0, 1, 0}
```

In order to check more random results, it is more efficient to first calculate the output state right before the measurement.

```
In[9]:= Timing[new = ExpressionFor[qc0];]
Out[9]= {3.15309, Null}
```

From the above output state, we perform measurements (as many times as we like) to get random results.

```
In[10]:= out = Measurement[new, S@cc];
phi = FromDigits[Readout[out, S@{cc}], 2] / $N
Out[10]= 11
          16
```

From the above random result, we determine the period of the function by utilizing the continued fraction representation of the estimated phase.

```
In[11]:= cnt = ContinuedFraction[phi];
candidates = Convergents[cnt]

Out[11]= {0, 1, 2, 5}

Out[12]= {0, 1, 2, 5}
          2
          3 , 11
          16}
```

From the above convergents, we have two candidates 3 (and its multiples 6, 9, 12, 15 are also candidates) and 16. Evaluation of the function for some arguments confirms that 3 is the period of the function.

4.5.2 The Order-Finding Algorithm

In the previous section, the function f was not specified explicitly, but the corresponding quantum oracle (4.82) was assumed to be implemented efficiently. Here we consider a specific function, the *modular exponentiation*

$$f(x) = a^x \pmod{N} \quad (4.94)$$

for fixed positive integers a and N . The two integers a and N are assumed to be coprimes. The function is a periodic function,

$$a^r = 1 \pmod{N} \quad (4.95)$$

for some integer r . The primary period (the smallest period) r is called the order of a modulo N . The order-finding problem is to determine the order for given a and N . It is known to be hard to solve classically. The order-finding algorithm is the central part of the quantum factorization algorithm to be discussed in Section 4.5.3.

Unlike the period-finding algorithm, the order-finding algorithm does not resort to quantum oracle, and implement the function with elementary gates. More

specifically, it uses the quantum phase estimation algorithm to estimate the phase of the unitary operator

$$\hat{U} : |x\rangle \mapsto |ax \pmod{N}\rangle , \quad (4.96)$$

the quantum mechanical extension of the *modular multiplication*.

A repeated application of the modular multiplication gives the modular exponentiation

$$\hat{U}^y : |x\rangle \mapsto |a^y x \pmod{N}\rangle . \quad (4.97)$$

As such, applying the modular multiplication r times should bring the system back to the original state. Therefore, the states

$$|1\rangle, \hat{U}|1\rangle, \hat{U}^2|1\rangle, \dots, \hat{U}^{r-1}|1\rangle \quad (4.98)$$

form a closed cycle. In physics, it is common to construct the eigenstates of \hat{U} that generates the cycle with the discrete Fourier transform of the states in the list.^{4.14} By definition, the latter is the quantum Fourier transform—see Eq. (4.38). Indeed, define

$$|\phi_k\rangle := \frac{1}{\sqrt{r}} \sum_{x=0}^{r-1} \hat{U}^r |1\rangle e^{2\pi i x k / r} = \frac{1}{\sqrt{r}} \sum_{x=0}^{r-1} |a^x \pmod{N}\rangle e^{2\pi i x k / r} \quad (4.99)$$

for $k = 0, 1, \dots, r - 1$. We see that

$$\hat{U}|\phi_k\rangle = \frac{1}{\sqrt{r}} \sum_{x=0}^{r-1} |a^{x+1} \pmod{N}\rangle e^{2\pi i x k / r} = e^{-2\pi i k / r} |\phi_k\rangle . \quad (4.100)$$

In other words, $|\phi_k\rangle$ is an eigenstate of \hat{U} belonging to the eigenvalue $e^{-2\pi i k / r}$. Therefore, by preparing the target register in one of the eigenstates $|\phi_k\rangle$ and applying the standard quantum phase estimation procedure, one would be able to estimate the “phase” $2\pi\phi_k := 2\pi k / r$. One would then find the order r from it.

Unfortunately, it is highly non-trivial to prepare a register in one of those eigenstates. Instead, we prepare the target register in a superposition of *all* eigenstates. It is also a common dictum in physics that a localized distribution in real space is unlocalized in the Fourier space and *vice versa*. Indeed, we find that

$$\sum_{k=0}^{r-1} |\phi_k\rangle = |1\rangle \equiv |0\rangle \otimes \dots \otimes |0\rangle \otimes |1\rangle . \quad (4.101)$$

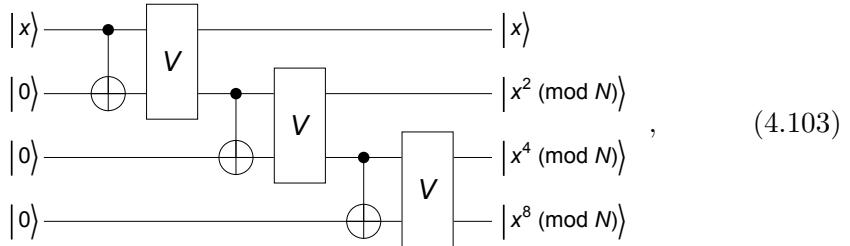
The quantum phase estimation procedure randomly selects one from the possible values $0, 1/r, \dots, (r-1)/r$. Further, the estimated phase ϕ is only approximate to one of the precise values $\phi_k = k/r$. We have already seen this situation in the period-finding algorithm. The (classical) post-processing is exactly the same as in the period-finding algorithm.

^{4.14}In condensed matter physics, the resulting states are called *Bloch states*.

There still remains a question when it comes to the performance of the order-finding algorithm. The performance of the algorithm relies on how efficiently one can perform the controlled- \hat{U}^{2^j} gate—see Section 4.4.2. As mentioned above in (4.97), \hat{U}^y is nothing but the modular exponentiation. Fortunately, it is known that the modular exponentiation can be implemented efficiently on a quantum computer (Vedral *et al.*, 1996). Let \hat{V} be the modular multiplication on *two* registers (to be compared with \hat{U} with fixed a in (4.96)),

$$\hat{V} : |x\rangle \otimes |y\rangle \mapsto |x\rangle \otimes |xy \text{ (mod } N\rangle . \quad (4.102)$$

The basic idea is to duplicate the logical state, $|x\rangle \otimes |0\rangle \rightarrow |x\rangle \otimes |x\rangle$, employing the CNOT gate and then apply the modular multiplication on the two registers, $|x\rangle \otimes |x\rangle \rightarrow |x^2 \text{ (mod } N\rangle$. Repeating the procedure j times produces the desired result. For example, the following quantum circuit model effectively performs \hat{U}^{2^3} ,



where each horizontal line denotes a *register* of n qubits, not a single qubit, and $x = 0, 1, \dots, 2^n - 1$.

4.5.3 Quantum Factorization Algorithm

The quantum factorization algorithm is composed of two parts, one quantum mechanical and the other classical. The quantum mechanical part is just the order-finding algorithm. The classical part is to determine the factors from the order based on number theory.

...
...

4.5.4 Quantum Search Algorithm

The quantum search algorithm—also known as *Grover's algorithm* after the inventor (Grover, 1996, 1997)—is a quantum algorithm for *unstructured* search. The unstructured search problem is to find a small number of elements with a designated property in a huge *unsorted* (or, more generally, *unstructured*) data. The quantum search algorithm finds a desired element with high probability with an order of \sqrt{N} queries to the function that defines the property, where N is the size of the data. Classical algorithms need N queries. Interestingly, the

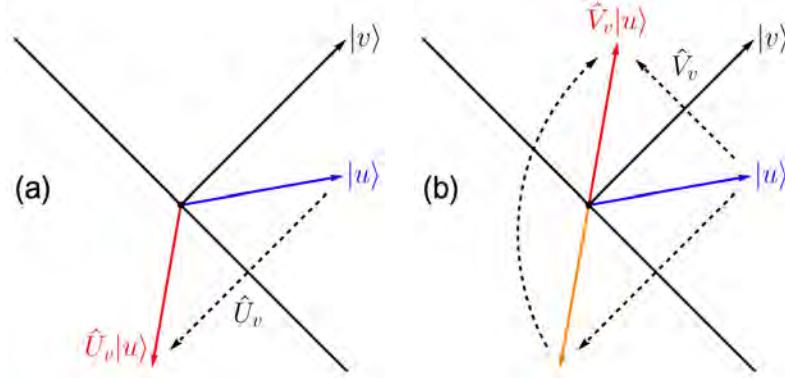


Figure 4.4: Geometric interpretation of (a) Householder transformation and (b) Grover's diffusion operator. (a) The Householder transformation \hat{U}_v associated with a normalized vector $|v\rangle$ is a reflection in the hyperplane containing the origin and perpendicular to $|v\rangle$. (b) Grover's diffusion operator \hat{V}_v with respect to ω is a reflection in the line along the vector $|v\rangle$. It can also be regarded as the Householder transformation \hat{U}_v associated with the same vector $|v\rangle$ followed by another reflection through the origin.

quantum search algorithm is known to be already optimal in the sense that any quantum algorithm for unstructured search needs an order of \sqrt{N} queries. In any case, the superiority of the quantum algorithm over its classical counterparts is merely quadratic (i.e., polynomial) rather than exponential. However, the quadratic speedup cannot be underestimated. The efficacy is considerable for large N . Further, the quantum search algorithm can apply to a wide range of problems. For example, it enables us to determine the existence and number of solutions of so-called NP-complete problems and to search exhaustively for the solutions.

To understand and analyze the quantum search algorithm, it is convenient to introduce two closely related unitary operators. One is a *Householder transformation*. Let $|v\rangle$ be a normalized vector in the Hilbert space. The Householder transformation associated with $|v\rangle$ is a unitary transformation defined by

$$\hat{U}_v := \hat{I} - 2|v\rangle\langle v|. \quad (4.104)$$

Geometrically, it is a reflection in the hyperplane perpendicular to the vector $|v\rangle$ and containing the origin (null vector) as illustrated in Fig. 4.4 (a). For this reason, it is also known as *Householder reflection*. The other is Grover's diffusion operator. Grover's diffusion operator associated with $|v\rangle$ is defined by

$$\hat{V}_v := 2|v\rangle\langle v| - \hat{I} = -\hat{U}_v. \quad (4.105)$$

It is a reflection in the line along the vector $|v\rangle$ as shown in Fig. 4.4 (b). It can also be regarded as the Householder reflection \hat{U}_v associated with the same vector $|v\rangle$ followed by another reflection through the origin—see Fig. 4.4 (b).

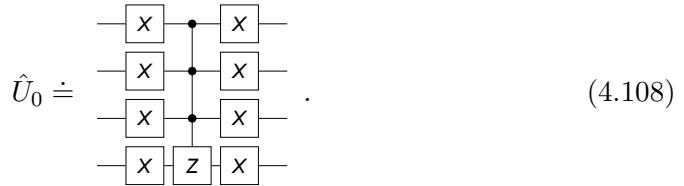
One can directly implement the Householder reflection \hat{U}_v for a known state $|v\rangle$. Suppose that $|v\rangle$ is achieved from $|0\rangle \equiv |0\rangle^{\otimes n}$ by a unitary transformation \hat{W} , $|v\rangle = \hat{W}|0\rangle$.^{4.15} Then it follows that

$$\hat{U}_v = \hat{W} \left(\hat{I} - 2|0\rangle\langle 0| \right) \hat{W}^\dagger = \hat{W}\hat{U}_0\hat{W}^\dagger. \quad (4.106)$$

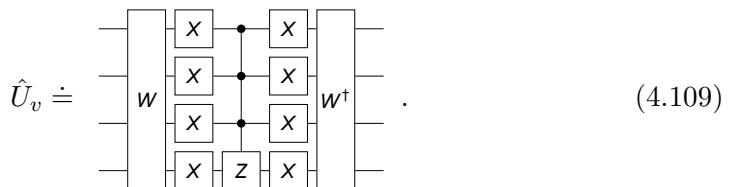
Note that \hat{U}_0 flips the phase of $|0\rangle$ to $-|0\rangle$ but leaves other states unaffected. It is equivalent to writing

$$\hat{U}_0 : |x_1 \cdots x_{n-1}\rangle \otimes |x_n\rangle \mapsto \begin{cases} |x_1 \cdots x_{n-1}\rangle \otimes (-\hat{Z}|x_n\rangle) & x_1 = \cdots = x_{n-1} = 0; \\ |x_1 \cdots x_{n-1}\rangle \otimes |x_n\rangle & (\text{otherwise}). \end{cases} \quad (4.107)$$

It is a variant of multi-qubit controlled- \hat{Z} gate. More explicitly, \hat{U}_0 for four qubits is described by the following quantum circuit



Overall, the Householder transformation associated with $|v\rangle$ can be implemented by a quantum circuit of the form



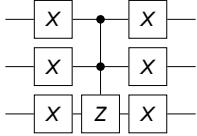
One can achieve Grover's diffusion operator \hat{V}_v for a known vector v in the same manner (up to a global phase factor -1).

Here is a quantum circuit model for Grover's diffusion operator.

We first implement $(1 - 2|\theta\rangle\langle\theta|)$.

^{4.15}Recall that there always exists such a unitary operator \hat{W} . See Theorem A.16.

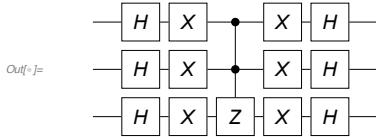
```
In[5]:= $n = 3;
jj = Range[$n];
ss = S[jj, None];
cz = CZ @@ Subsets[ss, {2}];
cz = Successive[cz, ss];
cz = CZ[S[1], #] & /@ Rest@ss;
qc = QuantumCircuit[S[jj, 1],
ControlledU[S[Most@jj], S[$n, 3]],
S[jj, 1]]
```



```
In[6]:= bs = Basis[S[jj]];
out = qc ** bs;
Thread[bs > out] // LogicalForm // TableForm
Out[6]/TableForm=
|0s10s20s3⟩ → -|0s10s20s3⟩
|0s10s21s3⟩ → |0s10s21s3⟩
|0s11s20s3⟩ → |0s11s20s3⟩
|0s11s21s3⟩ → |0s11s21s3⟩
|1s10s20s3⟩ → |1s10s20s3⟩
|1s10s21s3⟩ → |1s10s21s3⟩
|1s11s20s3⟩ → |1s11s20s3⟩
|1s11s21s3⟩ → |1s11s21s3⟩
```

This is Grover's diffusion operator (up to an irrelevant global phase factor of -1).

```
In[7]:= qc1 = QuantumCircuit[S[jj, 6], qc, S[jj, 6]]
```



Let us now describe the unstructured search problem more precisely. Suppose that we have a large list of items labelled by $x = 0, 1, 2, \dots, N - 1$. Among the items, there are a small number M ($1 \leq M \ll N$) of items with a unique property. The problem is to seek those items. The property is usually described by a classical oracle, $f : \{0, 1\}^n \rightarrow \{0, 1\}$, where we assume $N \equiv 2^n$. The desired items are designated by the condition $f(x) = 1$. In other words, an unstructured search is to find the solutions to the equation $f(x) = 1$. This is why the quantum search algorithm applies to a broad class of problems where a function $f(x)$ is well defined and can be evaluated without heavy cost.

We construct the quantum search algorithm combining Householder reflections and Grover's diffusion operators. Let \mathcal{M} be the set of the solutions,

$$\mathcal{M} := \{x : f(x) = 1, x = 0, 1, \dots, 2^n - 1\}. \quad (4.110)$$

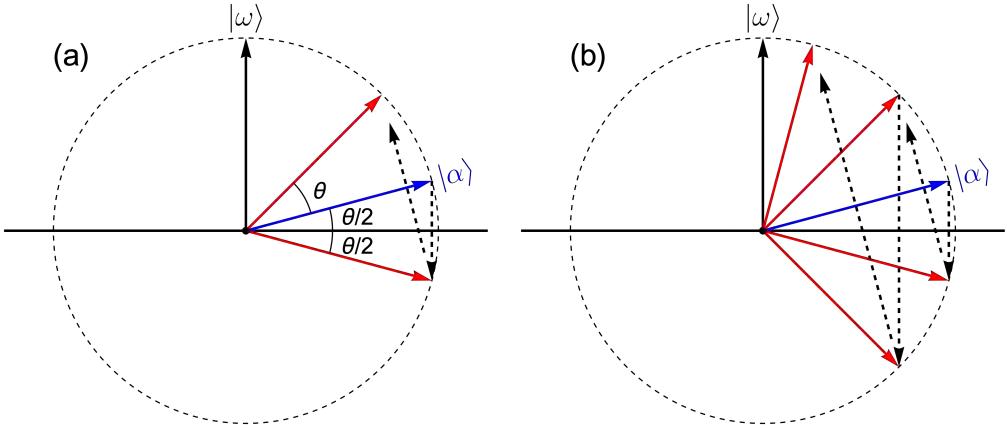


Figure 4.5: Geometric interpretation of the quantum search algorithm. (a) The application of the Householder reflection \hat{U}_ω followed by Grover's diffusion operator \hat{V}_α leads to the Grover rotation, $\hat{G} := \hat{V}_\alpha \hat{U}_\omega$, a rotation by angle 2θ . θ is the angle between $|\alpha\rangle$ and the horizontal axis. (b) Alternating applications of \hat{U}_ω and \hat{V}_α , or equivalently repeated applications of the Grover rotation \hat{G} , bring $|\alpha\rangle$ increasingly closer to $|\omega\rangle$.

Define a state of superposition consisting of the solutions

$$|\omega\rangle := \frac{1}{\sqrt{M}} \sum_{x \in \mathcal{M}} |x\rangle, \quad (4.111)$$

and another consisting of items that are not solutions

$$|\omega_\perp\rangle := \frac{1}{\sqrt{N-M}} \sum_{x \notin \mathcal{M}} |x\rangle. \quad (4.112)$$

Obviously, they are orthogonal to each other, $\langle \omega | \omega_\perp \rangle = 0$. Neither of the two states are known and can be prepared. Rather, $|\omega\rangle$ is the state we want to produce through the algorithm. Now consider the overall-superposition state, consisting all logical basis states

$$|\alpha\rangle := \frac{1}{\sqrt{N}} \sum_{x=0}^{N-1} |x\rangle. \quad (4.113)$$

As we have seen several times, this state can be prepared by applying the Hadamard gates $\hat{H}^{\otimes n}$ on $|0\rangle$. Since $|\alpha\rangle$ involves both solutions and non-solution, one can rewrite it into the form

$$|\alpha\rangle = \sqrt{\frac{M}{N}} |\omega\rangle + \sqrt{\frac{N-M}{N}} |\omega_\perp\rangle. \quad (4.114)$$

The advantage of the overall-superposition state $|\alpha\rangle$ is that one can make use of the linearity of quantum mechanics—or quantum parallelism—to evaluate an

operator on every state all at once. Indeed, applying the Householder reflection \hat{U}_ω associated with the state vector $|\omega\rangle$ on $|\alpha\rangle$ produces

$$\hat{U}_\omega |\alpha\rangle = -\sqrt{\frac{M}{N}} |\omega\rangle + \sqrt{\frac{N-M}{N}} |\omega_\perp\rangle = \frac{1}{\sqrt{N}} \sum_{x=0}^{N-1} |x\rangle (-1)^{f(x)}. \quad (4.115)$$

With a single operation of \hat{U}_ω , we have flagged the states $|x\rangle$ associated with the solutions by the reversed phase factor -1 . Unfortunately, however, we cannot directly single out those states. Instead, the tactic is to amplify the component along the state $|\omega\rangle$. This can be achieved by applying Grover's diffusion operator \hat{V}_α associated with $|\alpha\rangle$, $\hat{U}_\omega |\alpha\rangle \rightarrow \hat{V}_\alpha \hat{U}_\omega |\alpha\rangle$. As illustrated geometrically in Fig. 4.5 (a), it brings the state closer to the state $|\omega\rangle$.

At this stage, it is convenient to define the *Grover rotation* operator

$$\hat{G} := \hat{V}_\alpha \hat{U}_\omega. \quad (4.116)$$

\hat{G} describes a rotation within the subspace spanned by $|\omega\rangle$ and $|\alpha\rangle$ towards the desired state $|\omega\rangle$. For a more quantitative analysis, we define the angle θ by

$$\cos(\theta/2) = \sqrt{\frac{N-M}{N}}, \quad \sin(\theta/2) = \sqrt{\frac{M}{N}} \quad (4.117)$$

so that

$$|\alpha\rangle = |\omega\rangle \sin(\theta/2) + |\omega_\perp\rangle \cos(\theta/2). \quad (4.118)$$

It corresponds to the state rotated from $|\omega_\perp\rangle$ by angle θ around the axis perpendicular to both $|\omega\rangle$ and $|\omega_\perp\rangle$. Accordingly, \hat{G} corresponds to a rotation by angle 2θ around the same axis. This is illustrated in Fig. 4.5 (a). The k applications of the Grover rotation \hat{G} give

$$|\psi_k\rangle = \hat{G}^k |\alpha\rangle = |\omega\rangle \sin((k+1/2)\theta) + |\omega_\perp\rangle \cos((k+1/2)\theta). \quad (4.119)$$

Starting from $|\alpha\rangle$ in (4.118), the resulting state $|\psi_k\rangle$ gets closest to the desired state $|\omega\rangle$ when $(k+1/2)\theta \approx \pi/2$. Therefore, the optimal number K of the Grover rotations to apply is given by

$$K = \text{round}(\pi/2\theta - 1/2). \quad (4.120)$$

Assuming that $M \ll N$, we observe from (4.117) that

$$\theta \approx 2 \sin^{-1} \left(\sqrt{M/N} \right) \approx 2\sqrt{M/N} \ll 1. \quad (4.121)$$

It follows that

$$K \approx \frac{\pi}{4} \sqrt{\frac{N}{M}}. \quad (4.122)$$

It means that we can find the solution by applying the Grover rotations \hat{G} an order of \sqrt{N} times.

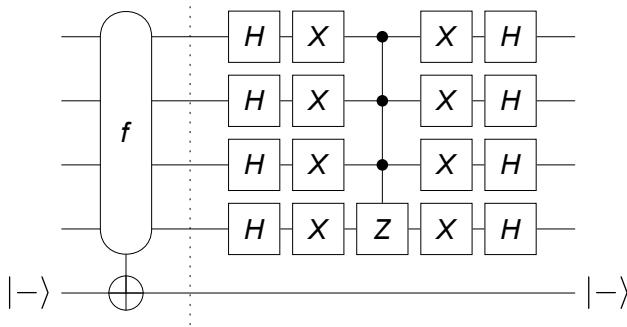


Figure 4.6: The implementation of the Grover rotation in a quantum circuit model. The first part before the vertical dashed line effectively implements the Householder transformation \hat{U}_ω . The second part corresponds to Grover's diffusion operator \hat{V}_α .

We can make the estimation of the performance more rigorous and get the upper bound. We note that

$$K \leq \text{ceil}(\pi/2\theta). \quad (4.123)$$

In accordance with (4.117), we note that

$$\theta/2 \geq \sin(\theta/2) = \sqrt{M/N}, \quad (4.124)$$

where we have assumed that $M \leq N/2$. Putting it to (4.123), we find that

$$K \leq \frac{\pi}{4} \sqrt{\frac{N}{M}}. \quad (4.125)$$

In short, the estimation in (4.122) turns out to be the worst case estimation.

In the above, we have described the quantum search algorithm in terms of the Grover rotation consisting of the Householder reflection \hat{U}_ω and Grover's diffusion operator \hat{V}_α . As we have noted in (4.109), \hat{V}_α can be implemented in terms of elementary gates because the state $|\alpha\rangle$ is known. However, $|\omega\rangle$ is unknown, and one cannot implement \hat{U}_ω in a similar way. Fortunately, we have noted in (4.115) that the effect of \hat{U}_ω is flagging the states $|x\rangle$ corresponding to the solutions with the reversed phase factor -1 . One can achieve exactly the same effect by employing the quantum oracle (Section 4.2.1) corresponding to the classical oracle $f(x)$. In particular, with the ancilla qubit prepared in the state $|-\rangle := (|0\rangle - |1\rangle)/\sqrt{2}$, the quantum oracle induces a conditional phase shift as follows [see Eq. (4.19) and (4.20)]

$$|x\rangle \otimes |-\rangle \mapsto \left(|x\rangle (-1)^{f(x)} \right) \otimes |-\rangle. \quad (4.126)$$

It is illustrated in the quantum circuit model as follows

$$\begin{array}{c} |x\rangle \left\{ \begin{array}{c} \text{---} \\ \text{---} \\ \text{---} \\ \text{---} \\ f \\ \text{---} \\ \text{---} \end{array} \right\} |x\rangle (-1)^{f(x)} \\ = |x\rangle \left\{ \begin{array}{c} \text{---} \\ \text{---} \\ \text{---} \\ \text{---} \\ f \\ \text{---} \\ \text{---} \end{array} \right\} \hat{U}_\omega |x\rangle . \end{array} \quad (4.127)$$

|---> |--->

The overall quantum circuit model to implement the Grover rotation is depicted in Fig. 4.6.

Here we demonstrate the quantum search algorithm.

We work with 4 (system) qubits and one ancillary qubit. The ancillary qubit is for the quantum oracle.

```
$n = 4;
cc = Range[$n - 1];
tt = $n;
jj = Range[$n];
aa = $n + 1;
```

Here we specify a classical oracle. The function f is more human readable. The function g will be used in the actual quantum oracle.

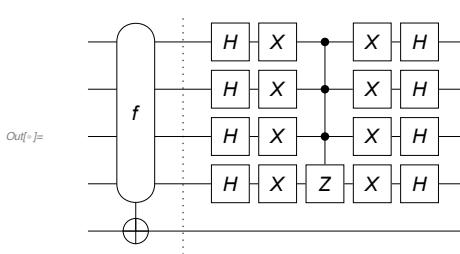
```
f[3] = 1;
f[_] = 0;
g[x_] := f@FromDigits[{x}, 2]
```

The ancilla qubit is prepared in the following state.

```
In[7]:= in = ProductState[S[aa] → {1, -1} / Sqrt[2]];
in // Elaborate // LogicalForm
Out[7]=  $\frac{|0_{S_5}\rangle}{\sqrt{2}} - \frac{|1_{S_5}\rangle}{\sqrt{2}}$ 
```

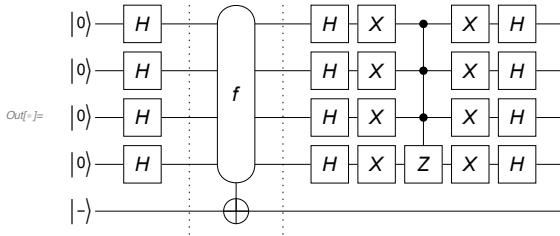
This is a quantum circuit for the Grover rotation.

```
In[8]:= grv = QuantumCircuit[
  Oracle[g, S[jj], S[aa]], "Separator",
  S[jj, 6], S[jj, 1], ControlledU[S[cc], S[tt, 3]], S[jj, 1], S[jj, 6],
  "PortSize" → {0.5, 0.2}
]
```



We apply the Grover rotation once on the input state.

```
In[6]:= qc1 = QuantumCircuit[LogicalForm[Ket[], S[jj]], {in, "Label" -> "| - >"}, S[jj, 6], "Separator", grv]
```

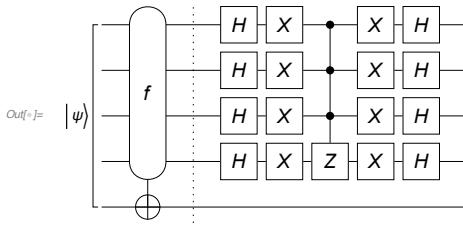


Here is the result of it.

```
In[7]:= out1 = Elaborate[qc1]
Out[7]= -\frac{3|-\rangle}{16\sqrt{2}} - \frac{3|1_{S_1}\rangle}{16\sqrt{2}} - \frac{3|1_{S_1}1_{S_2}\rangle}{16\sqrt{2}} - \frac{3|1_{S_1}1_{S_2}1_{S_3}\rangle}{16\sqrt{2}} - \frac{3|1_{S_1}1_{S_2}1_{S_3}1_{S_4}\rangle}{16\sqrt{2}} + \frac{3|1_{S_1}1_{S_2}1_{S_3}1_{S_4}1_{S_5}\rangle}{16\sqrt{2}} + \frac{3|1_{S_1}1_{S_2}1_{S_3}1_{S_5}\rangle}{16\sqrt{2}} - \frac{3|1_{S_1}1_{S_2}1_{S_4}\rangle}{16\sqrt{2}} + \frac{3|1_{S_1}1_{S_2}1_{S_4}1_{S_5}\rangle}{16\sqrt{2}} + \frac{3|1_{S_1}1_{S_2}1_{S_5}\rangle}{16\sqrt{2}} - \frac{3|1_{S_1}1_{S_3}\rangle}{16\sqrt{2}} - \frac{3|1_{S_1}1_{S_3}1_{S_4}\rangle}{16\sqrt{2}} + \frac{3|1_{S_1}1_{S_3}1_{S_4}1_{S_5}\rangle}{16\sqrt{2}} + \frac{3|1_{S_1}1_{S_3}1_{S_5}\rangle}{16\sqrt{2}} - \frac{3|1_{S_1}1_{S_4}\rangle}{16\sqrt{2}} + \frac{3|1_{S_1}1_{S_4}1_{S_5}\rangle}{16\sqrt{2}} + \frac{3|1_{S_1}1_{S_5}\rangle}{16\sqrt{2}} - \frac{3|1_{S_2}\rangle}{16\sqrt{2}} - \frac{3|1_{S_2}1_{S_3}\rangle}{16\sqrt{2}} + \frac{3|1_{S_2}1_{S_3}1_{S_4}\rangle}{16\sqrt{2}} - \frac{3|1_{S_2}1_{S_3}1_{S_5}\rangle}{16\sqrt{2}} + \frac{3|1_{S_2}1_{S_4}\rangle}{16\sqrt{2}} - \frac{3|1_{S_2}1_{S_4}1_{S_5}\rangle}{16\sqrt{2}} + \frac{3|1_{S_2}1_{S_5}\rangle}{16\sqrt{2}} - \frac{11|1_{S_3}1_{S_4}\rangle}{16\sqrt{2}} - \frac{3|1_{S_3}\rangle}{16\sqrt{2}} + \frac{11|1_{S_3}1_{S_4}1_{S_5}\rangle}{16\sqrt{2}} + \frac{3|1_{S_3}1_{S_5}\rangle}{16\sqrt{2}} - \frac{3|1_{S_4}\rangle}{16\sqrt{2}} + \frac{3|1_{S_4}1_{S_5}\rangle}{16\sqrt{2}} + \frac{3|1_{S_5}\rangle}{16\sqrt{2}}
```

We apply the Grover rotation again on the above output state.

```
In[8]:= qc2 = QuantumCircuit[{out1, "Label" -> "| \psi >"}, grv]
```



```
In[5]:= out2 = Elaborate[qc2]
Out[5]= 
$$\frac{5 |\_ \rangle}{64 \sqrt{2}} - \frac{5 |1_{S_1}1_{S_2}1_{S_3}1_{S_4}1_{S_5}\rangle}{64 \sqrt{2}} + \frac{5 |1_{S_1}\rangle}{64 \sqrt{2}} + \frac{5 |1_{S_1}1_{S_2}\rangle}{64 \sqrt{2}} + \frac{5 |1_{S_1}1_{S_2}1_{S_3}\rangle}{64 \sqrt{2}} +$$


$$\frac{5 |1_{S_1}1_{S_2}1_{S_3}1_{S_4}\rangle}{64 \sqrt{2}} - \frac{5 |1_{S_1}1_{S_2}1_{S_3}1_{S_5}\rangle}{64 \sqrt{2}} - \frac{5 |1_{S_1}1_{S_2}1_{S_4}1_{S_5}\rangle}{64 \sqrt{2}} + \frac{5 |1_{S_1}1_{S_2}1_{S_4}\rangle}{64 \sqrt{2}} -$$


$$\frac{5 |1_{S_1}1_{S_2}1_{S_5}\rangle}{64 \sqrt{2}} + \frac{5 |1_{S_1}1_{S_3}\rangle}{64 \sqrt{2}} + \frac{5 |1_{S_1}1_{S_3}1_{S_4}\rangle}{64 \sqrt{2}} - \frac{5 |1_{S_1}1_{S_3}1_{S_4}1_{S_5}\rangle}{64 \sqrt{2}} - \frac{5 |1_{S_1}1_{S_3}1_{S_5}\rangle}{64 \sqrt{2}} +$$


$$\frac{5 |1_{S_1}1_{S_4}\rangle}{64 \sqrt{2}} - \frac{5 |1_{S_1}1_{S_4}1_{S_5}\rangle}{64 \sqrt{2}} - \frac{5 |1_{S_1}1_{S_5}\rangle}{64 \sqrt{2}} - \frac{5 |1_{S_2}1_{S_3}1_{S_4}1_{S_5}\rangle}{64 \sqrt{2}} + \frac{5 |1_{S_2}\rangle}{64 \sqrt{2}} + \frac{5 |1_{S_2}1_{S_3}\rangle}{64 \sqrt{2}} +$$

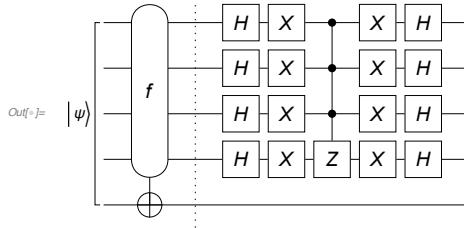

$$\frac{5 |1_{S_2}1_{S_3}1_{S_4}\rangle}{64 \sqrt{2}} - \frac{5 |1_{S_2}1_{S_3}1_{S_5}\rangle}{64 \sqrt{2}} + \frac{5 |1_{S_2}1_{S_4}\rangle}{64 \sqrt{2}} - \frac{5 |1_{S_2}1_{S_4}1_{S_5}\rangle}{64 \sqrt{2}} - \frac{5 |1_{S_2}1_{S_5}\rangle}{64 \sqrt{2}} -$$


$$\frac{61 |1_{S_3}1_{S_4}1_{S_5}\rangle}{64 \sqrt{2}} - \frac{5 |1_{S_3}1_{S_5}\rangle}{64 \sqrt{2}} + \frac{5 |1_{S_3}\rangle}{64 \sqrt{2}} + \frac{61 |1_{S_3}1_{S_4}\rangle}{64 \sqrt{2}} - \frac{5 |1_{S_4}1_{S_5}\rangle}{64 \sqrt{2}} + \frac{5 |1_{S_4}\rangle}{64 \sqrt{2}} - \frac{5 |1_{S_5}\rangle}{64 \sqrt{2}}$$

```

We repeat the same procedure.

```
In[6]:= qc3 = QuantumCircuit[{out2, "Label" \rightarrow "|\psi\rangle"}, grv]
```



```
In[7]:= out3 = Elaborate[qc3]
Out[7]= 
$$\frac{13 |\_ \rangle}{256 \sqrt{2}} - \frac{13 |1_{S_1}1_{S_2}1_{S_3}1_{S_4}1_{S_5}\rangle}{256 \sqrt{2}} + \frac{13 |1_{S_1}\rangle}{256 \sqrt{2}} + \frac{13 |1_{S_1}1_{S_2}\rangle}{256 \sqrt{2}} + \frac{13 |1_{S_1}1_{S_2}1_{S_3}\rangle}{256 \sqrt{2}} +$$


$$\frac{13 |1_{S_1}1_{S_2}1_{S_3}1_{S_4}\rangle}{256 \sqrt{2}} - \frac{13 |1_{S_1}1_{S_2}1_{S_3}1_{S_5}\rangle}{256 \sqrt{2}} - \frac{13 |1_{S_1}1_{S_2}1_{S_4}1_{S_5}\rangle}{256 \sqrt{2}} + \frac{13 |1_{S_1}1_{S_2}1_{S_4}\rangle}{256 \sqrt{2}} -$$


$$\frac{13 |1_{S_1}1_{S_2}1_{S_5}\rangle}{256 \sqrt{2}} + \frac{13 |1_{S_1}1_{S_3}\rangle}{256 \sqrt{2}} + \frac{13 |1_{S_1}1_{S_3}1_{S_4}\rangle}{256 \sqrt{2}} - \frac{13 |1_{S_1}1_{S_3}1_{S_4}1_{S_5}\rangle}{256 \sqrt{2}} - \frac{13 |1_{S_1}1_{S_3}1_{S_5}\rangle}{256 \sqrt{2}} +$$


$$\frac{13 |1_{S_1}1_{S_4}\rangle}{256 \sqrt{2}} - \frac{13 |1_{S_1}1_{S_4}1_{S_5}\rangle}{256 \sqrt{2}} - \frac{13 |1_{S_1}1_{S_5}\rangle}{256 \sqrt{2}} - \frac{13 |1_{S_2}1_{S_3}1_{S_4}1_{S_5}\rangle}{256 \sqrt{2}} + \frac{13 |1_{S_2}\rangle}{256 \sqrt{2}} + \frac{13 |1_{S_2}1_{S_3}\rangle}{256 \sqrt{2}} +$$

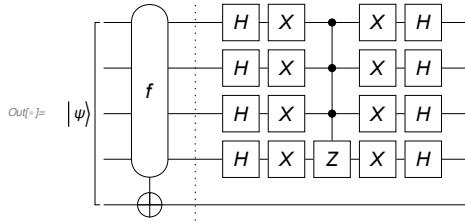

$$\frac{13 |1_{S_2}1_{S_3}1_{S_4}\rangle}{256 \sqrt{2}} - \frac{13 |1_{S_2}1_{S_3}1_{S_5}\rangle}{256 \sqrt{2}} + \frac{13 |1_{S_2}1_{S_4}\rangle}{256 \sqrt{2}} - \frac{13 |1_{S_2}1_{S_4}1_{S_5}\rangle}{256 \sqrt{2}} - \frac{13 |1_{S_2}1_{S_5}\rangle}{256 \sqrt{2}} -$$


$$\frac{251 |1_{S_3}1_{S_4}\rangle}{256 \sqrt{2}} + \frac{13 |1_{S_3}\rangle}{256 \sqrt{2}} + \frac{251 |1_{S_3}1_{S_4}1_{S_5}\rangle}{256 \sqrt{2}} - \frac{13 |1_{S_3}1_{S_5}\rangle}{256 \sqrt{2}} - \frac{13 |1_{S_4}1_{S_5}\rangle}{256 \sqrt{2}} + \frac{13 |1_{S_4}\rangle}{256 \sqrt{2}} - \frac{13 |1_{S_5}\rangle}{256 \sqrt{2}}$$

```

In this case, the optimal number of applications of the Grover algorithm is four. So, this is supposed to be the last step.

```
In[7]:= qc4 = QuantumCircuit[{out3, "Label" -> "|\psi>"}, grv]
```



This is the final state.

```
In[8]:= out4 = Elaborate[qc4]
```

$$\begin{aligned} \text{Out[8]= } & -\frac{171|_>}{1024\sqrt{2}} - \frac{171|1_{S_1}>}{1024\sqrt{2}} - \frac{171|1_{S_1}1_{S_2}>}{1024\sqrt{2}} - \frac{171|1_{S_1}1_{S_2}1_{S_3}>}{1024\sqrt{2}} - \frac{171|1_{S_1}1_{S_2}1_{S_3}1_{S_4}>}{1024\sqrt{2}} + \\ & \frac{171|1_{S_1}1_{S_2}1_{S_3}1_{S_4}1_{S_5}>}{1024\sqrt{2}} + \frac{171|1_{S_1}1_{S_2}1_{S_3}1_{S_5}>}{1024\sqrt{2}} - \frac{171|1_{S_1}1_{S_2}1_{S_4}>}{1024\sqrt{2}} + \frac{171|1_{S_1}1_{S_2}1_{S_4}1_{S_5}>}{1024\sqrt{2}} + \\ & \frac{171|1_{S_1}1_{S_3}>}{1024\sqrt{2}} - \frac{171|1_{S_1}1_{S_3}>}{1024\sqrt{2}} - \frac{171|1_{S_1}1_{S_3}1_{S_4}>}{1024\sqrt{2}} + \frac{171|1_{S_1}1_{S_3}1_{S_4}1_{S_5}>}{1024\sqrt{2}} + \\ & \frac{171|1_{S_1}1_{S_3}1_{S_5}>}{1024\sqrt{2}} - \frac{171|1_{S_1}1_{S_4}>}{1024\sqrt{2}} + \frac{171|1_{S_1}1_{S_4}1_{S_5}>}{1024\sqrt{2}} + \frac{171|1_{S_1}1_{S_5}>}{1024\sqrt{2}} - \frac{171|1_{S_2}>}{1024\sqrt{2}} - \\ & \frac{171|1_{S_2}1_{S_3}>}{1024\sqrt{2}} - \frac{171|1_{S_2}1_{S_3}>}{1024\sqrt{2}} + \frac{171|1_{S_2}1_{S_3}1_{S_4}>}{1024\sqrt{2}} + \frac{171|1_{S_2}1_{S_3}1_{S_5}>}{1024\sqrt{2}} - \\ & \frac{171|1_{S_2}1_{S_4}>}{1024\sqrt{2}} + \frac{171|1_{S_2}1_{S_4}>}{1024\sqrt{2}} + \frac{171|1_{S_2}1_{S_5}>}{1024\sqrt{2}} - \frac{781|1_{S_3}1_{S_4}1_{S_5}>}{1024\sqrt{2}} - \frac{171|1_{S_3}>}{1024\sqrt{2}} + \\ & \frac{781|1_{S_3}1_{S_4}>}{1024\sqrt{2}} + \frac{171|1_{S_3}1_{S_5}>}{1024\sqrt{2}} - \frac{171|1_{S_4}>}{1024\sqrt{2}} + \frac{171|1_{S_4}1_{S_5}>}{1024\sqrt{2}} + \frac{171|1_{S_5}>}{1024\sqrt{2}} \end{aligned}$$

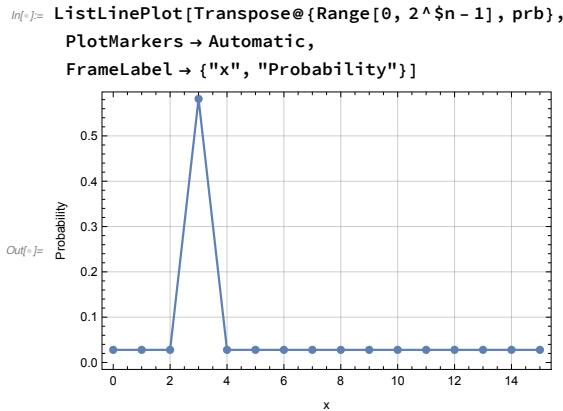
To analyze the result, we ignore the ancillary qubit. Note that the result is a density matrix.

```
new = Matrix@PartialTrace[out4, S[aa]];
```

The diagonal elements of the density matrix are the probability in the logical basis.

```
prb = Diagonal[new];
```

As you can see below, the probability for $x=3$ is the highest. We conclude that the quantum search algorithm works reasonably well.



Problems

- 4.1. Consider the quantum oracle defined in (4.12).
- Classically (operating only on the basis states without any superposition of them), the mapping in (4.11) is one-to-one regardless of the function f .
 - Show that for any function f , the transformation \hat{U}_f in (4.12) is unitary.
- 4.2. **(conditional phase shift)** Let $f : \{0, 1\}^n \rightarrow \{0, 1\}$ be a (classical) oracle. Suppose that we have a quantum computer consisting of an n -qubit “control register” and a single-qubit “target register”. Using a quantum oracle, construct a quantum circuit model which shifts the phase by the factor $e^{i\phi}$ of every term in $|x\rangle$ satisfying $f(x) = 1$ of the n -qubit register, but keeps the second single-qubit register intact. The quantum circuit model effectively transforms the states of the control qubit as

$$\sum_x |x\rangle \mapsto \sum_x |x\rangle e^{i\phi f(x)}. \quad (4.128)$$

The simple application of quantum oracle as in Eq. (4.19) corresponds to $\phi = \pi$.

Hint: See the implementation of the controlled- U gate in Section 2.2.2.

- 4.3. Prove the identity

$$e^{-ia\hat{P}} \hat{X} e^{+ia\hat{P}} = \hat{X} - a \quad (4.129)$$

in Eq. (4.43).

Hint: See Appendix A.4.2, and recall that $e^{-ia\hat{P}} = \sum_y |P_y\rangle e^{-iap_y} \langle P_y|$.

4.4. Using the orthogonality relation

$$\sum_{z=0}^{2^n-1} e^{i(x-y)p_z} = 2^n \delta_{xy} \quad (4.130)$$

for all $x, y = 0, 1, \dots, 2^n - 1$, prove the identity

$$\hat{U}_{\text{QFT}}^\dagger |X_y\rangle = \frac{1}{2^{n/2}} \sum_x |X_x\rangle e^{-ixp_y} \quad (4.131)$$

in Eq. (4.61).

4.5. Consider the logical state $|X_0\rangle \equiv |0\rangle^{\otimes n}$ of an n -qubit register. Show that

$$\hat{U}_{\text{QFT}} |X_0\rangle = \hat{U}_{\text{QFT}}^\dagger |X_0\rangle = \hat{H}^{\otimes n} |0\rangle^{\otimes n} = \frac{1}{2^{n/2}} \sum_{x=0}^{2^n-1} |X_x\rangle = |P_0\rangle. \quad (4.132)$$

Chapter 5

Decoherence

- August 2, 2021 (v1.16)

In the previous chapters, our discussions and arguments have been mainly based on the principles of quantum physics for closed systems. However, no realistic system is closed. A system is naturally subject to interaction with the surrounding system, which is commonly called the *environment*. There is also a more fundamental reason for the notion of an *open quantum system* in quantum mechanics. The theory of quantum mechanics is intrinsically probabilistic. It means that the verification of any quantum principle should be tested statistically through repeated measurements and the resulting data. The measurement process inevitably requires coupling the system to a measuring device. Further, in quantum computation and more general quantum information processing, we desire the preparation, manipulation, and measurement of quantum states. All those procedures require the system to be coupled to external equipment.

In principle, one can regard the combined system enclosing both the system and the environment as a closed system, and apply the quantum mechanical principles to the total system. However, the environment is typically large—since perfect isolation is impossible, the total system is eventually the whole universe—and involves a huge number of degrees of freedom. A complete microscopic description incorporating the environmental degrees of freedom is not only impractical but also of little use. First of all, such a description is tremendously complicated and hard to solve. A solution, if any, would lead to an intractable amount of information, the vast majority of which would be irrelevant to the physical effects exhibited by the system itself.

A more reasonable and practical approach is thus to seek an effective description of open quantum systems in terms of the system degrees of freedom only. The effective theory is achieved in two stages: First, the ignorance of the environmental degrees of freedom brings about the statistical mixture of pure states for the system. The state of the system is not a pure state any longer and is described by the density operator. We have already introduced this description in Section 1.1.2.

Second, the influence of the environment should be reflected on the (effective) dynamical evolution of the density operator in a way that does not depend on the details of the environment and the system-environment coupling. A powerful mathematical tool is provided by the formalism of quantum operations.

In this chapter, we first introduce quantum operations formalism. The two common and complementary representations of quantum operations are discussed together with simple examples. Quantum operations are used not only for dynamical processes of open quantum systems but also for the quantum theory of generalized measurement. Next, we will turn to the quantum master equation approach to open quantum systems. It is an approximate approach of the quantum operations formalism under the Markovian assumption. While quantum operations formalism provides the most general mathematical tool, it is not always possible to find the quantum operations explicitly for given specific systems. It is far simpler and insightful to construct the quantum master equation and examine the solution to understand the behaviors of the open quantum systems in question. In the remaining part of the chapter, we introduce several concepts such as entropy and fidelity to quantify and characterize quantum information. These information-theoretic concepts will be useful in the next chapter, where we discuss the quantum error correction code.

5.1 Quantum Operations

Under a certain physical process, the state of a given system evolves into another state. The time evolution of a closed system is described by unitary operators. What about an open quantum system, which interacts with its environment?

Dynamical processes of open quantum systems are described by a special kind of supermaps (Appendix B.2) called *quantum operations*: A supermap transforms density operators to other density operators while preserving the elementary properties of density operators. In particular, as density operators are positive,^{5.1} a quantum operation needs to preserve positivity. However, it turns out that merely preserving positivity is not sufficient and a much stronger condition is required. Essentially, a quantum operation needs to preserve not only the positivity of density operators of a given system but also all density operators of any extended system including the system itself and its surrounding systems. Mathematically, such a condition is satisfied by *completely positive* supermaps (Definition B.4).

Let us define quantum operations more precisely. Let \mathcal{V} and \mathcal{W} be vector spaces. Suppose that \mathcal{F} is a supermap from $\mathcal{L}(\mathcal{V})$ onto $\mathcal{L}(\mathcal{W})$. \mathcal{F} is called a *quantum operation* if it satisfies the following three axioms

^{5.1}Recall that a positive operator is Hermitian by definition.

(a) \mathcal{F} never increases the trace. That is, $0 \leq \mathcal{F}(\hat{\rho}) \leq 1$ for any *density operator*^{5.2} $\hat{\rho}$ on \mathcal{V} .

(b) \mathcal{F} is *convex linear*. That is, for any probabilities p_j ^{5.3} and density operators $\hat{\rho}_j$ on \mathcal{V} ,

$$\mathcal{F}\left(\sum_j \hat{\rho}_j p_j\right) = \sum_j \mathcal{F}(\hat{\rho}_j) p_j. \quad (5.1)$$

(c) \mathcal{F} is a *completely positive supermap*.^{5.4} That is, not only $\mathcal{F}(\hat{\rho})$ itself is positive for any positive operator $\hat{\rho}$ on \mathcal{V} , but $(\mathcal{F} \otimes \mathcal{I})(\hat{\rho})$ is also positive for any positive operator on $\mathcal{V} \otimes \mathcal{E}$ with arbitrary vector space \mathcal{E} .

Most quantum operations preserve the trace— $\text{Tr } \mathcal{F}(\hat{\rho}) = 1$ for all density operators $\hat{\rho}$. An important exception is the process associated with a (generalized selective) measurement. When the trace is not preserved, $\text{Tr } \mathcal{F}(\hat{\rho})$ gives the probability for the dynamical process \mathcal{F} to occur.

In quantum information theory, quantum operations preserving trace, i.e., completely positive and trace-preserving supermaps, are called *quantum channels*. Physically, they describe communication channels that can transmit quantum information, as well as classical information.

Another important class of physical phenomena described by quantum operations is *quantum decoherence* or just *decoherence* for short, referring to the loss of quantum coherence: Consider a quantum state and its representation in a certain basis.^{5.5} The components of the representation are complex numbers in general. As long as there exists a definite phase relation between different components, the state is said to be coherent. For various reasons, which are eventually traced back to interaction with the environment, the state loses coherence and the quantum effects disappear in the system. In this case, the input and output Hilbert space coincide, $\mathcal{V} = \mathcal{W}$, and the relevant quantum operations are *superoperators*.

In this section, we introduce three mathematical methods to describe quantum operations, the Kraus representation in Section 5.1.1, the Choi isomorphism in Section 5.1.2, and the unitary representation in Section 5.1.3. We conclude this section by giving examples of these representations for single-qubit systems in Section 5.1.4.

^{5.2}That is, $\hat{\rho}$ is positive semi-definite and $\text{Tr } \hat{\rho} = 1$. See 1.1.2 for the precise definition and properties of a density operator.

^{5.3}That is, $0 \leq p_j \leq 1$ and $\sum_j p_j = 1$.

^{5.4}In most literature, it is called a completely positive “map”.

^{5.5}It is important to note that coherence (and hence decoherence) is a basis-dependent concept.

5.1.1 Kraus Representation

A quantum operation is a restricted form of completely positive supermap. Accordingly, for any quantum operation \mathcal{F} from $\mathcal{L}(\mathcal{V})$ onto $\mathcal{L}(\mathcal{W})$, there exist linear maps \hat{F}_μ from \mathcal{V} onto \mathcal{W} such that

$$\mathcal{F}(\hat{\rho}) = \sum_{\mu} \hat{F}_{\mu} \hat{\rho} \hat{F}_{\mu}^{\dagger} \quad (5.2)$$

for all linear operators (not necessarily density operators) $\hat{\rho}$ on \mathcal{V} (Theorem B.5). The linear maps \hat{F}_μ are called the *Kraus elements* or the *Kraus maps* of \mathcal{F} . The trace-decreasing condition in axiom (a) imposes the inequalities

$$0 \leq \sum_{\mu} \hat{F}_{\mu}^{\dagger} \hat{F}_{\mu} \leq 1. \quad (5.3)$$

One can always choose Kraus elements that are *mutually orthogonal* with respect to the trace Hermitian product, that is,

$$\text{Tr } \hat{F}_{\mu}^{\dagger} \hat{F}_{\nu} = 0 \quad (5.4)$$

whenever $\mu \neq \nu$. Through the procedure of choosing orthogonal Kraus elements, one can drastically optimize Kraus elements.

The Kraus representation of a quantum operation in a sum of operator provides powerful tools to analyse the quantum operation. But, at this stage, Kraus representation follows from a mathematical theorem for completely positive map (Theorem B.5). How does Kraus representation arise physically? What are the physical meanings of the Kraus elements? These questions are the subject of this subsection. We will also discuss some basic properties of Kraus representation.

To see how Kraus representation arises physically, let us consider a system interacting with its environment. We denote by \mathcal{V} and \mathcal{E} the Hilbert spaces associated with the system and the environment, respectively. For simplicity, we assume that the total system is initially in the product state $\hat{\rho} \otimes \hat{\sigma}$. The total system is a closed system, and the dynamical process afterwards due to the system-environment interaction is described an overall unitary operator \hat{U} acting on the total system, $\hat{\rho} \otimes \hat{\sigma} \mapsto \hat{U}(\hat{\rho} \otimes \hat{\sigma})\hat{U}^{\dagger}$. Without access to the environment, one has to take the partial trace of the final state over the environment to obtain the state of the system. Putting all together, the quantum operation \mathcal{F} describing the process is written as

$$\mathcal{F}(\hat{\rho}) = \text{Tr}_{\mathcal{E}} \hat{U}(\hat{\rho} \otimes \hat{\sigma})\hat{U}^{\dagger} = \sum_{\mu} \langle \varepsilon_{\mu} | \hat{U}(\hat{\rho} \otimes \hat{\sigma})\hat{U}^{\dagger} | \varepsilon_{\mu} \rangle , \quad (5.5)$$

where $\{|\varepsilon_{\mu}\rangle\}$ is an orthonormal basis of \mathcal{E} . On the right-hand side of (5.5), the Hermitian product is applied partially and only on \mathcal{E} , and the expression still

remains to be an operator on \mathcal{V} . To further investigate the quantum operation, we take the spectral decomposition (see Appendix A.4) of $\hat{\sigma}$

$$\hat{\sigma} = \sum_{\nu} |s_{\nu}\rangle \langle s_{\nu}| , \quad (5.6)$$

where the eigenvectors $|s_{\mu}\rangle$ have been normalized by their own eigenvalues $s_{\mu} = \langle s_{\mu}|s_{\mu}\rangle$ —see Eq. (A.43)—as $\hat{\sigma}$ is a positive semidefinite operator. Now, define linear maps $\hat{F}_{\mu} : \mathcal{V} \rightarrow \mathcal{W}$ by

$$\hat{F}_{\mu} := \sum_{\nu} \text{Tr}_{\mathcal{E}} \left(\hat{U} \left(\hat{I} \otimes |s_{\nu}\rangle \langle \varepsilon_{\mu}| \right) \right) = \sum_{\nu} \langle \varepsilon_{\mu}| \hat{U} |s_{\nu}\rangle . \quad (5.7)$$

Physically, \hat{F}_{μ} describes the dynamics of the system under the condition that the environment has made a transition to the state $|\varepsilon_{\mu}\rangle$. They satisfy the closure relation

$$\sum_{\mu} \hat{F}_{\mu}^{\dagger} \hat{F}_{\mu} = \hat{I} . \quad (5.8)$$

Putting (5.7) into (5.5), we arrive at the representation

$$\mathcal{F}(\hat{\rho}) = \sum_{\mu} \hat{F}_{\mu} \hat{\rho} \hat{F}_{\mu}^{\dagger} \quad (5.9)$$

for the quantum operation.

Let us take a toy model for a specific example. Consider a chain of three qubits. Suppose that the Hamiltonian of the chain is given by

$$\hat{H} = \frac{1}{2} B \hat{S}_1^z + \frac{1}{2} J \left(\hat{S}_1^x \hat{S}_2^x + \hat{S}_2^x \hat{S}_3^x \right) . \quad (5.10)$$

We regard the first qubit as the “system”, and the other two qubits form the “environment”. The coupling constant J indicates how strong the system interacts with the environment—without the coupling J , the system is a closed system, and the evolution of its quantum state should be unitary. This model is overly artificial as the environment is not only finite but also very small, just twice larger than the system. However, it is enough to demonstrate the main idea. We consider the whole chain is initially in the product state $|\Psi(0)\rangle = |L\rangle \otimes |L\rangle \otimes |L\rangle$, where $|L\rangle := (|0\rangle + i|1\rangle)/\sqrt{2}$ is the “left” state—it is often used to denote the left-circularly polarized state of a photon. When focused on the system only, the initial state is $|L\rangle$, or equivalently, $\hat{\rho}(0) = |L\rangle \langle L| = \frac{1}{2} \hat{I}_1 + \frac{1}{2} \hat{S}_1^y$. At later time t , the state of the chain is given by

$$|\Psi(t)\rangle = \hat{U}(t) |\Psi(0)\rangle , \quad (5.11)$$

where the unitary operator $\hat{U}(t) = \exp(-it\hat{H})$ governs the evolution of the total system (the chain). Ignoring the two qubits in the environment, we get the (mixed) state of the system

$$\hat{\rho}(t) = \text{Tr}_{\mathcal{E}} |\Psi(t)\rangle \langle \Psi(t)| = \frac{1}{2} \hat{I}_1 - \frac{1}{2} \frac{\sin(\Omega t)}{\Omega/B} \hat{S}_1^x + \frac{1}{2} \cos(\Omega t) \hat{S}_1^y , \quad (5.12)$$

where $\Omega := \sqrt{B^2 + J^2}$. Now, let us describe the evolution $\hat{\rho}(0) \rightarrow \hat{\rho}(t)$ in terms of quantum operation \mathcal{F}_t depending parametrically on time t . More specifically, we want to find the Kraus elements $\hat{F}_\mu(t)$ for the quantum operation so that

$$\hat{\rho}(t) = \sum_\mu \hat{F}_\mu(t) \hat{\rho}(0) \hat{F}_\mu^\dagger(t). \quad (5.13)$$

Following the prescription in (5.7), we can get the Kraus elements

$$\hat{F}_\mu(t) = \text{Tr}_{\mathcal{E}} \hat{U}(t) \left(\hat{I} \otimes |L\rangle \langle \mu_1| \otimes |L\rangle \langle \mu_2| \right), \quad (5.14)$$

where μ_j are the binary digits of $\mu = (\mu_1\mu_2)_2$. More explicitly, they are given by

$$\hat{F}_0(t) = \frac{\cos(\frac{\Omega t}{2}) \hat{I}_1}{2} - \frac{iB \sin(\frac{\Omega t}{2}) \hat{S}_1^z}{2\Omega} + \frac{Je^{-iJt} \sin(\frac{\Omega t}{2}) \hat{S}_1^x}{2\Omega}, \quad (5.15a)$$

$$\hat{F}_1(t) = \frac{\cos(\frac{\Omega t}{2}) \hat{I}_1}{2} - \frac{iB \sin(\frac{\Omega t}{2}) \hat{S}_1^z}{2\Omega} + \frac{Je^{+iJt} \sin(\frac{\Omega t}{2}) \hat{S}_1^x}{2\Omega}, \quad (5.15b)$$

$$\hat{F}_2(t) = \frac{\cos(\frac{\Omega t}{2}) \hat{I}_1}{2} - \frac{iB \sin(\frac{\Omega t}{2}) \hat{S}_1^z}{2\Omega} - \frac{Je^{+iJt} \sin(\frac{\Omega t}{2}) \hat{S}_1^x}{2\Omega}, \quad (5.15c)$$

$$\hat{F}_3(t) = \frac{\cos(\frac{\Omega t}{2}) \hat{I}_1}{2} - \frac{iB \sin(\frac{\Omega t}{2}) \hat{S}_1^z}{2\Omega} - \frac{Je^{-iJt} \sin(\frac{\Omega t}{2}) \hat{S}_1^x}{2\Omega}. \quad (5.15d)$$

With direct evaluations, one can convince oneself that these Kraus elements indeed reproduce the dynamical evolution in (5.13), and that $\sum_\mu \hat{F}_\mu^\dagger(t) \hat{F}_\mu(t) = \hat{I}$. The Kraus elements in (5.15) are not mutually orthogonal—and they can be optimized as we will see below. In particular, all the Kraus elements are identical,

$$\hat{F}_0(t) = \hat{F}_1(t) = \hat{F}_2(t) = \hat{F}_3(t) = \frac{1}{2} \cos\left(\frac{Bt}{2}\right) \hat{I}_1 - \frac{i}{2} \sin\left(\frac{Bt}{2}\right) \hat{S}_1^z, \quad (5.16)$$

in the absence of the coupling ($J = 0$). In this case, the quantum operation is described by a single Kraus element

$$\hat{F}(t) := 2\hat{F}_0^\dagger(t)\hat{I}_1 = \cos(Bt/2) - i \sin(Bt/2) \hat{S}_1^z = \exp\left(-i\hat{S}^z Bt/2\right), \quad (5.17)$$

which is obviously unitary as it should.

We consider a chain of three qubits. The first qubit is regarded as the “system”, and the other two form the “environment”.

```
$L = 3;
jj = Range[$L];
sys = 1;
env = Range[2, $L];
```

Here is the Hamiltonian describing the chain. We are measuring the energy in units of B ($B=1$).

```
In[5]:= Let[Real, J]
H = S[sys, 3] / 2 + J / 2 * Total[ChainBy[S[jj, 1], Multiply]]
Out[5]=  $\frac{1}{2} J (S_1^x S_2^x + S_2^x S_3^x) + \frac{S_1^z}{2}$ 
```

The system has a discrete symmetry. It is invariant under the rotation around the z-axis by angle π .

```
In[6]:= V = Rotation[Pi, S[1, 3]] ** Rotation[Pi, S[2, 3]] ** Rotation[Pi, S[3, 3]]
V ** H ** Dagger[V]
Out[6]=  $i S_1^z S_2^z S_3^z$ 
Out[6]=  $\frac{1}{2} J S_1^x S_2^x + \frac{1}{2} J S_2^x S_3^x + \frac{S_1^z}{2}$ 
```

The symmetry leads to the degeneracy of the eigenvalues of the Hamiltonian.

```
In[7]:= ProperValues[H]
Out[7]=  $\left\{ \frac{1}{2} (-J - \sqrt{1+J^2}), \frac{1}{2} (-J - \sqrt{1+J^2}), \frac{1}{2} (J - \sqrt{1+J^2}), \frac{1}{2} (J - \sqrt{1+J^2}), \frac{1}{2} (-J + \sqrt{1+J^2}), \frac{1}{2} (-J + \sqrt{1+J^2}), \frac{1}{2} (J + \sqrt{1+J^2}), \frac{1}{2} (J + \sqrt{1+J^2}) \right\}$ 
```

The time-evolution operator of the chain is evaluated.

```
Let[Real, t]
U[t_] = Elaborate@MultiplyExp[-I t H];
```

We suppose that the chain is initially in the state $|L\rangle \otimes |L\rangle \otimes |L\rangle$, where $|L\rangle = (|0\rangle + i|1\rangle)/\sqrt{2}$.

```
In[8]:= vec[0] = ProductState[S@jj → {1, I} / Sqrt[2]]
vec[t_] = U[t] ** Elaborate[vec[0]];
Out[8]=  $\left( \frac{|0\rangle}{\sqrt{2}} + \frac{i|1\rangle}{\sqrt{2}} \right)_{S_1} \otimes \left( \frac{|0\rangle}{\sqrt{2}} + \frac{i|1\rangle}{\sqrt{2}} \right)_{S_2} \otimes \left( \frac{|0\rangle}{\sqrt{2}} + \frac{i|1\rangle}{\sqrt{2}} \right)_{S_3}$ 
```

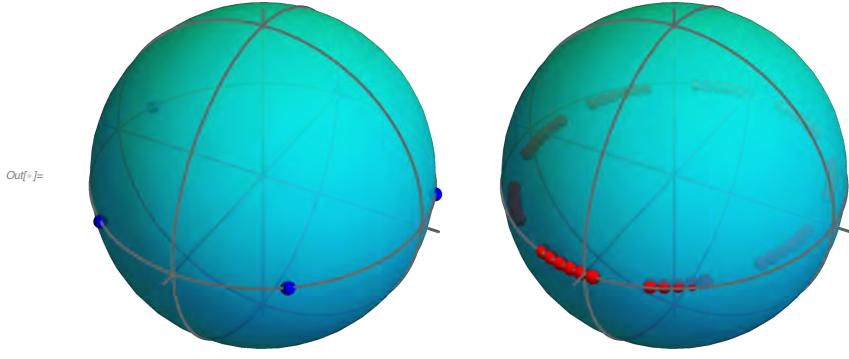
Here is a set of replacement rules to be used later to simplify expressions.

```
In[9]:= rules = {Sqrt[1 + J^2] → Ω, 1 / Sqrt[1 + J^2] → 1 / Ω}
Out[9]=  $\left\{ \sqrt{1+J^2} \rightarrow \Omega, \frac{1}{\sqrt{1+J^2}} \rightarrow \frac{1}{\Omega} \right\}$ 
```

```
In[10]:= rho[t_] = PartialTrace[vec[t], S@env] // Elaborate // ExpToTrig // Garner;
rho[t] /. rules
Out[10]=  $\frac{1}{2} + \frac{1}{2} \cos[t\Omega] S_1^y - \frac{S_1^x \sin[t\Omega]}{2\Omega}$ 
```

Take a look at the dynamical evolution of the state of the “system” (the first qubit), after tracing out the “environment” (the other two qubits). On the left, shown is the evolution in the absence of the coupling to the environment. On the right, the evolution is not coherent due to the coupling to the environment. The evolution is still periodic because the environment is finite.

```
In[7]:= bv0 = Block[{J = 0.}, Table[BlochVector[rho[2 Pi * t]], {t, 0, 5, 0.2}]];
bv = Block[{J = .5}, Table[BlochVector[rho[2 Pi * t]], {t, 0, 5/J, 0.2}]];
GraphicsRow@{BlochSphere[{Blue, Bead /@ bv0}], BlochSphere[{Red, Bead /@ bv}]}
```



Now, let us examine the evolution in terms of the supermap. This is the initial state of the system.

```
In[8]:= in = Elaborate@ProductState[S@sys → {1, I} / Sqrt[2]];
in = Elaborate@Dyad[in, in]
Out[8]=  $\frac{1}{2} + \frac{S_1^y}{2}$ 
```

To find the Kraus elements, consider the initial state of the environment.

```
In[9]:= sgm = Elaborate@ProductState[S@env → {1, I} / Sqrt[2]];
sgm // LogicalForm
Out[9]=  $\frac{1}{2} |\Theta_{S_2}\Theta_{S_3}\rangle + \frac{1}{2} i |\Theta_{S_2}1_{S_3}\rangle + \frac{1}{2} i |1_{S_2}\Theta_{S_3}\rangle - \frac{1}{2} |1_{S_2}1_{S_3}\rangle$ 
```

Finally, this is the Kraus elements of the quantum operation.

```
bs = Basis[S@env];
prj = Map[Dyad[sgm, #, S@env] &, bs];
ops = U[t] ** prj // Elaborate;

In[10]:= kraus = PartialTrace[#, S@env] & /@ ops // ExpToTrig // Elaborate // Garner;
kraus /. rules
Out[10]= 
$$\begin{aligned} & \left\{ \frac{1}{2} \cos\left[\frac{t\Omega}{2}\right] \times \left( \cos\left[\frac{3t}{2}\right] + i \sin\left[\frac{3t}{2}\right] \right) + \frac{\Im S_1^x \left( \cos\left[\frac{3t}{2}\right] - i \sin\left[\frac{3t}{2}\right] \right) \times \sin\left[\frac{t\Omega}{2}\right]}{2\Omega} + \right. \\ & \frac{S_1^z \left( -i \cos\left[\frac{3t}{2}\right] + \sin\left[\frac{3t}{2}\right] \right) \sin\left[\frac{t\Omega}{2}\right]}{2\Omega}, \frac{1}{2} \cos\left[\frac{t\Omega}{2}\right] \left( i \cos\left[\frac{3t}{2}\right] + \sin\left[\frac{3t}{2}\right] \right) + \\ & \frac{S_1^z \left( \cos\left[\frac{3t}{2}\right] - i \sin\left[\frac{3t}{2}\right] \right) \times \sin\left[\frac{t\Omega}{2}\right]}{2\Omega} + \frac{i \Im S_1^x \left( \cos\left[\frac{3t}{2}\right] + i \sin\left[\frac{3t}{2}\right] \right) \times \sin\left[\frac{t\Omega}{2}\right]}{2\Omega}, \\ & \frac{1}{2} \cos\left[\frac{t\Omega}{2}\right] \left( i \cos\left[\frac{3t}{2}\right] + \sin\left[\frac{3t}{2}\right] \right) + \frac{\Re S_1^z \left( \cos\left[\frac{3t}{2}\right] - i \sin\left[\frac{3t}{2}\right] \right) \times \sin\left[\frac{t\Omega}{2}\right]}{2\Omega} + \\ & \frac{\Im S_1^x \left( -i \cos\left[\frac{3t}{2}\right] + \sin\left[\frac{3t}{2}\right] \right) \sin\left[\frac{t\Omega}{2}\right]}{2\Omega}, -\frac{1}{2} \cos\left[\frac{t\Omega}{2}\right] \times \left( \cos\left[\frac{3t}{2}\right] + i \sin\left[\frac{3t}{2}\right] \right) + \\ & \left. \frac{\Im S_1^z \left( \cos\left[\frac{3t}{2}\right] - i \sin\left[\frac{3t}{2}\right] \right) \times \sin\left[\frac{t\Omega}{2}\right]}{2\Omega} + \frac{i \Re S_1^x \left( \cos\left[\frac{3t}{2}\right] + i \sin\left[\frac{3t}{2}\right] \right) \times \sin\left[\frac{t\Omega}{2}\right]}{2\Omega} \right\} \end{aligned}$$

```

```
In[=]:= new[t_] = Elaborate@Supermap[kraus][in];
new[t] /. rules
Out[=]=  $\frac{1}{2} + \frac{1}{2} \cos[t\Omega] S_1^y - \frac{S_1^x \sin[t\Omega]}{2\Omega}$ 

In[=]:= new[t] - rho[t] // Elaborate // Garner
Out[=]= 0
```

In the above arguments, we have derived a Kraus representation for a quantum operation based on a system-plus-environment model. While it provides a useful physical picture of quantum operations, at first glance, the resulting representation does not look particularly useful. The Kraus elements \hat{F}_μ are not orthogonal—with respect to the trace Hermitian product in (B.4)—to each other. Even worse, the number of Kraus elements may be as huge as the dimension of the environmental Hilbert space \mathcal{E} —the dimension of \mathcal{E} is infinite for any realistic environment. However, neither raises a significant problem. A formal representation in the form of Kraus representation already facilitates analysis of the quantum operation. Further, one can optimize the Kraus elements by reconstructing orthogonal Kraus elements.

Given a set of Kraus elements, how can one actually choose new Kraus elements that are mutually orthogonal.^{5.6} Let $\{\hat{E}_\mu\}$ be a basis of the space $\mathcal{L}(\mathcal{V}, \mathcal{W})$ of all linear maps from \mathcal{V} to \mathcal{W} . We expand \hat{F}_ν in the basis

$$\hat{F}_\nu = \sum_\mu \hat{E}_\mu M_{\mu\nu}, \quad (5.18)$$

where M is the matrix of expansion coefficients. Putting it back to (5.2), we have

$$\mathcal{F}(\hat{\rho}) = \sum_{\mu\nu} \hat{E}_\mu \hat{\rho} \hat{E}_\nu^\dagger \left(MM^\dagger \right)_{\mu\nu}. \quad (5.19)$$

The square matrix MM^\dagger of size $(\dim \mathcal{V} \dim \mathcal{W})$ is positive semidefinite, and can be decomposed into $MM^\dagger = V\Lambda V^\dagger$, where V is a unitary matrix and Λ is a diagonal matrix with all elements non-negative. We now define new Kraus elements

$$\hat{F}'_\nu := \sum_\mu \hat{E}_\mu \left(V\sqrt{\Lambda} \right)_{\mu\nu}. \quad (5.20)$$

Then, it is clear that they are mutually orthogonal. Further,

$$\mathcal{F}(\hat{\rho}) = \sum_{\mu=0}^{N-1} \hat{F}'_\mu \hat{\rho} \hat{F}'_\mu^\dagger \quad (5.21)$$

where $N \leq \dim \mathcal{V} \dim \mathcal{W}$. Therefore, it is noted that a set of mutually orthogonal Kraus elements is optimal in the sense that it has no more elements than $(\dim \mathcal{V}) \times$

^{5.6}It can also be seen in the lines to Theorem B.3. See also Eqs. (5.23) and (5.24).

$(\dim \mathcal{W})$. Applying this method to the previous example, we can get a new set of mutually orthogonal Kraus elements

$$\hat{F}'_0(t) = \cos(\Omega t/2) - i(B/\Omega) \sin(\Omega t/2) \hat{S}_1^z, \quad (5.22a)$$

$$\hat{F}'_1(t) = (J/\Omega) \sin(\Omega t/2) \hat{S}_1^x, \quad (5.22b)$$

from the four Kraus elements in (5.15). The same quantum operation $\hat{\rho}(0) \rightarrow \hat{\rho}(t)$ is now specified just by two Kraus elements.

We close this subsection by noting that the Kraus representation is not unique—we have just seen that given a set of Kraus elements, we could find another set of Kraus elements that are orthogonal to each other. There exists unitary freedom for the choice of the Kraus elements. Suppose that the two quantum operations \mathcal{F} and \mathcal{G} are associated with the Kraus elements $\{\hat{F}_\mu\}$ and $\{\hat{G}_\nu\}$, respectively.^{5.7} Then, $\mathcal{F} = \mathcal{G}$, that is,

$$\sum_\mu \hat{F}_\mu \hat{\rho} \hat{F}_\mu^\dagger = \sum_\nu \hat{G}_\nu \hat{\rho} \hat{G}_\nu^\dagger \quad (5.23)$$

for all $\hat{\rho} \in \mathcal{L}(\mathcal{V})$ if and only if there exists a unitary matrix U —attaching rows or columns if necessary—such that

$$\hat{G}_\nu = \sum_\mu \hat{F}_\mu U_{\mu\nu}. \quad (5.24)$$

This is analogous to the unitary freedom for the choice of pure states in the specification of a mixed state—see Eqs. (1.12) and (1.13). In fact, the underlying mathematical principles are the same. As we have already established the proof of the unitary freedom in the mixed state, here let us use it to prove the unitary freedom in the Kraus representation.

If the two sets of Kraus elements satisfy the relation (5.24), it is straight forward to prove the two quantum operations are identical—it immediately follows from the defining property of a unitary matrix. Let us prove the converse. Suppose that $\mathcal{F} = \mathcal{G}$. Then the corresponding Choi operators—see Appendix B.2.3—should be identical as well, $\hat{C}_{\mathcal{F}} = \hat{C}_{\mathcal{G}}$. Given the Kraus elements, one can evaluate the Choi operators explicitly starting from the maximally entangled state in Eq. (5.31),

$$\hat{C}_{\mathcal{F}} = \sum_\mu (\hat{F}_\mu \otimes \hat{I}) |\Phi\rangle \langle \Phi| (\hat{F}_\mu \otimes \hat{I})^\dagger = \sum_\mu |F_\mu\rangle \langle F_\mu|, \quad (5.25a)$$

$$\hat{C}_{\mathcal{G}} = \sum_\nu (\hat{G}_\nu \otimes \hat{I}) |\Phi\rangle \langle \Phi| (\hat{G}_\nu \otimes \hat{I})^\dagger = \sum_\nu |G_\nu\rangle \langle G_\nu|, \quad (5.25b)$$

where $|F_\mu\rangle, |G_\nu\rangle \in \mathcal{W} \otimes \mathcal{V}$ are the Choi vectors—see Appendix B.2.3—corresponding to the linear maps \hat{F}_μ and \hat{G}_ν , respectively,

$$|F_\mu\rangle := (\hat{F}_\mu \otimes \hat{I}) |\Phi\rangle, \quad |G_\nu\rangle := (\hat{G}_\nu \otimes \hat{I}) |\Phi\rangle. \quad (5.26)$$

^{5.7}The Kraus elements here are not orthogonal, $\text{Tr } \hat{F}_\mu^\dagger \hat{F}_\nu \neq 0$ and $\text{Tr } \hat{G}_\mu^\dagger \hat{G}_\nu \neq 0$, in general.

According to Eqs. (1.12) and (1.13),

$$\sum_{\mu} |F_{\mu}\rangle \langle F_{\mu}| = \sum_{\nu} |G_{\nu}\rangle \langle G_{\nu}| \quad (5.27)$$

implies that there exists a unitary matrix U such that

$$|G_{\nu}\rangle = \sum_{\mu} |F_{\mu}\rangle U_{\mu\nu}. \quad (5.28)$$

Finally, we note—Appendix B.2.3—that for arbitrary $|\psi\rangle \in \mathcal{V}$

$$\hat{G}_{\nu} |\psi\rangle = \langle \psi^* | F_{\nu} \rangle = \sum_{\mu} \langle \psi^* | E_{\mu} \rangle U_{\mu\nu} = \sum_{\mu} \hat{F}_{\mu} |\psi\rangle U_{\mu\nu}. \quad (5.29)$$

This asserts the relation (5.24).

In some cases, say, motivated by the (unperturbed) Hamiltonian of the isolated system, there may be a preferred basis. Can we exploit the unitary freedom to change the given set of Kraus elements to another set of Kraus elements that is consistent with the preferred basis? Unfortunately, given two sets of Kraus elements, it is not trivial to check if they are equivalent or not. It is because the Kraus elements in the relation (5.24) are not normalized. In terms of the normalized Kraus elements, $\hat{F}'_{\mu} = \hat{F}_{\mu} \sqrt{p_{\mu}}$ and $\hat{G}'_{\nu} = \hat{G}_{\nu} \sqrt{q_{\nu}}$, the relation reads as

$$\hat{G}'_{\nu} = \sum_{\mu} \hat{F}'_{\mu} \sqrt{p_{\mu}} U_{\mu\nu} / \sqrt{q_{\nu}}. \quad (5.30)$$

As \hat{F}'_{μ} and \hat{G}'_{ν} are orthonormal, the matrix $\sqrt{p_{\mu}} U_{\mu\nu} / \sqrt{q_{\nu}}$ should also be unitary. In general, it is not trivial to find a unitary matrix U which allows $\sqrt{p_{\mu}} U_{\mu\nu} / \sqrt{q_{\nu}}$ to be unitary as well.

5.1.2 Choi Isomorphism

Choi isomorphism is a one-to-one correspondence between supermaps and (usual) operators (Appendix B.2.3). It allows us to inspect a supermap in terms of the corresponding operator. It can reveal additional properties of the supermap that is not immediately clear from the supermap itself. As such, Choi isomorphism is not confined to quantum operations. It applies to the whole space of supermaps. Interestingly, however, Choi isomorphism remains to hold between *completely positive supermap* and *density operators*.^{5.8} It makes Choi isomorphism so useful for the studies of quantum operations.

In this subsection, we introduce Choi isomorphism. We then use it to provide two physically-motivated proofs of the Kraus representation theorem summarized in (5.2).

^{5.8}Note that an isomorphism between two spaces does not necessarily hold between their subspaces.

To exploit the condition of \mathcal{F} being completely positive later, take a copy of the original vector space \mathcal{V} as a reference space (or any vector space \mathcal{R} of the same dimension as \mathcal{V}), and construct the tensor-product space $\mathcal{V} \otimes \mathcal{V}$ of the original and reference space. Then, consider a maximally entangled state

$$|\Phi\rangle := \sum_k |v_k\rangle \otimes |v_k\rangle , \quad (5.31)$$

where $\{|v_k\rangle\}$ is an orthonormal basis of \mathcal{V} . Its density operator is given by

$$|\Phi\rangle \langle \Phi| = \sum_{kl} |v_k\rangle \langle v_l| \otimes |v_k\rangle \otimes \langle v_l| . \quad (5.32)$$

Now operate an extended supermap $\mathcal{F} \otimes \mathcal{I}$, where \mathcal{I} is the identity superoperator, on $|\Phi\rangle \langle \Phi|$ to get

$$\hat{C}_{\mathcal{F}} := (\mathcal{F} \otimes \mathcal{I})(|\Psi\rangle \langle \Psi|) = \sum_{ij} \sum_{kl} |w_i v_k\rangle \langle w_j v_l| C_{ik;jl} \in \mathcal{L}(\mathcal{W} \otimes \mathcal{V}) , \quad (5.33)$$

where C is the *Choi matrix* (see Appendix B.2.3) associated with \mathcal{F} ,

$$\mathcal{F}(|v_k\rangle \langle v_l|) = \sum_{ij} |w_i\rangle \langle w_j| C_{ik;jl} . \quad (5.34)$$

Equation (5.33) implies that $\hat{C}_{\mathcal{F}}$ is an operator (not a superoperator) on $\mathcal{W} \otimes \mathcal{V}$ with the matrix representation given just by the Choi matrix C —the operator $\hat{C}_{\mathcal{F}}$ and the matrix C are essentially the same mathematical objects. We call $\hat{C}_{\mathcal{F}}$ the *Choi operator* associated with the supermap \mathcal{F} . The association $\mathcal{F} \mapsto \hat{C}_{\mathcal{F}}$ is an isomorphism between supermaps and operators, and called the *Choi isomorphism*. The Choi operator $\hat{C}_{\mathcal{F}}$ (and hence the Choi matrix C) completely characterizes the associated supermap \mathcal{F} . To see it in a more physically transparent way, it is useful to represent the Choi operator $\hat{C}_{\mathcal{F}}$ in a quantum circuit model of the form

$$\hat{C}_{\mathcal{F}} = |\Phi\rangle \left\{ \begin{array}{c} \text{---} \\ \boxed{\mathcal{F}} \\ \text{---} \end{array} \right. \quad (5.35)$$

The Choi operator $\hat{C}_{\mathcal{F}}$ is the result of the evolution of the maximally entangled state $|\Phi\rangle$ in $\mathcal{V} \otimes \mathcal{V}$, composed of the original and reference space, under the supermap \mathcal{F} acting only on the original space. One way to quantitatively characterize the supermap \mathcal{F} is thus to compare $\hat{C}_{\mathcal{F}}$ with the initial maximally entangled state $|\Phi\rangle$,

$$F_E := \text{Tr} |\Phi\rangle \langle \Phi| \hat{C}_{\mathcal{F}} = \langle \Phi| \hat{C}_{\mathcal{F}} |\Phi\rangle . \quad (5.36)$$

The quantitative measure F_E is called the *entanglement fidelity*^{5.9} of the supermap \mathcal{F} . It quantifies how well the supermap preserves the initial quantum information.

^{5.9}It is a special case of more general notion of *fidelity*. The fidelity between two pure states $|v\rangle$ and $|w\rangle$ is defined by $|\langle v|w\rangle|$.

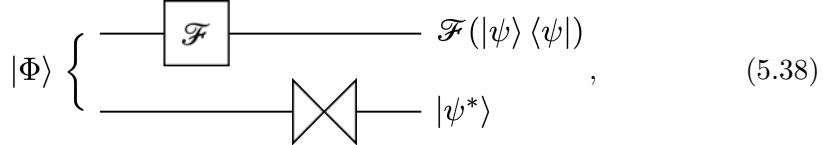
The Choi operator also reflects the properties of the associated supermap: When \mathcal{F} is completely positive, $\hat{C}_{\mathcal{F}}$ is a positive operator. Further, when the supermap \mathcal{F} is a *quantum operation*, satisfying all the axioms (a)–(c), $\hat{C}_{\mathcal{F}}$ is a density operator. Here it is interesting to note that the Choi isomorphism remains to hold between *quantum operations* and *density operators*. This refined isomorphism is called the *channel-state duality*.

Let us now prove the Kraus representation theorem using Choi isomorphism. It is clear that any supermap in the form (5.2) satisfies the three axioms, and is a quantum operation. The converse is more complicated to prove. There are three common ways, each of which is interesting in its own right. The first method—see Exercise B.5—is directly based on the general operator-sum representation in (B.17). Here we will discuss the other two methods.

The second method relies on the properties of the Choi operator $\hat{C}_{\mathcal{F}}$. For any pure state $|\psi\rangle = \sum_j |v_j\rangle \psi_j \in \mathcal{V}$, define its conjugate state $|\psi^*\rangle := \sum_j |v_j\rangle \psi_j^*$, and observe that

$$\mathcal{F}(|\psi\rangle \langle \psi|) = \text{Tr}_{\mathcal{V}} \left[\left(\hat{I} \otimes |\psi^*\rangle \langle \psi^*| \right) \hat{C}_{\mathcal{F}} \right] = \langle \psi^* | \hat{C}_{\mathcal{F}} | \psi^* \rangle. \quad (5.37)$$

As it happens often in this chapter, the Hermitian product in $\langle \psi^* | \hat{C}_{\mathcal{F}} | \psi^* \rangle$ is merely a short-hand notation for the partial trace, and $\langle \psi^* | \hat{C}_{\mathcal{F}} | \psi^* \rangle$ is an operator on \mathcal{W} (not a number). Recalling the quantum circuit representation (5.35) of the Choi isomorphism, the identity (5.37) can be described by the quantum circuit model



where the quantum circuit element $\text{---} \times \text{---}$ represents the projection onto the state specified at the output port. Since $\hat{C}_{\mathcal{F}}$ is a positive operator for a quantum operation \mathcal{F} , rewrite it in a spectral decomposition (see Appendix A.4, especially, Eq. (A.43))

$$\hat{C}_{\mathcal{F}} = \sum_{\mu} |\varphi_{\mu}\rangle \langle \varphi_{\mu}|, \quad (5.39)$$

where each vector $|\varphi_{\mu}\rangle$ has been normalized so that $\langle \varphi_{\mu} | \varphi_{\mu} \rangle$ gives the corresponding (positive) eigenvalue of $\hat{C}_{\mathcal{F}}$, $\hat{C}_{\mathcal{F}} |\varphi_{\mu}\rangle = |\varphi_{\mu}\rangle \langle \varphi_{\mu} | \varphi_{\mu} \rangle$. We define a linear map $\hat{F}_{\mu} : \mathcal{V} \rightarrow \mathcal{W}$ by the association

$$\hat{F}_{\mu} |\psi\rangle = \langle \psi^* | \varphi_{\mu} \rangle. \quad (5.40)$$

Note that on the right-hand side of the relation, the Hermitian product is applied partially and only on \mathcal{V} —the remaining part is a vector belonging to \mathcal{W} . Putting (5.39) and (5.40) into (5.37), we confirm that

$$\mathcal{F}(|\psi\rangle \langle \psi|) = \sum_{\mu} \hat{F}_{\mu} |\psi\rangle \langle \psi| \hat{F}_{\mu}^\dagger. \quad (5.41)$$

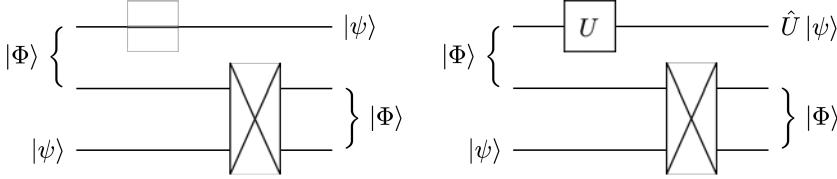
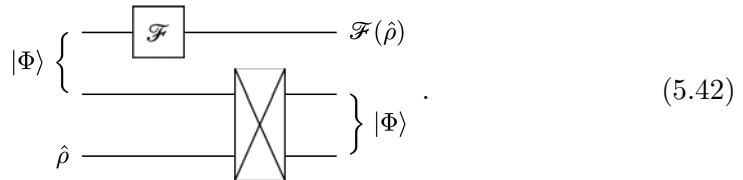


Figure 5.1: Comparison of quantum teleportation and quantum gate teleportation. (a) A simplified quantum circuit model of quantum teleportation. The Bell measurement is replaced by the projection to $|\Phi\rangle$, and the success probability is 1/4. (b) A quantum circuit model for quantum gate teleportation. The input state $|\psi\rangle$ on the third qubit results in the unitary-transformed state $\hat{U}|\psi\rangle$ on the first qubit.

As \mathcal{F} is linear and $|\psi\rangle$ is arbitrary, this proves the statement in the theorem.

Now, turn to the third proof. It is based on the so-called *quantum gate teleportation* protocol. Figure 5.1 (a) shows a simplified quantum circuit model of the quantum teleportation protocol. Compared with the typical quantum teleportation protocol discussed in Section 4.1, the Bell measurement has been replaced with the projection onto a single Bell state $|\Phi\rangle$. Due to the modification, the protocol is not deterministic any longer. Nevertheless, with the success probability 1/4, the input state $|\psi\rangle$ on the third qubit is “teleported” to the first qubit. At the end of the protocol, one can apply any unitary transformation \hat{U} to get $\hat{U}|\psi\rangle$. The result does not change if one applies \hat{U} even before the projection. This variation leads to the quantum circuit model depicted in Fig. 5.1 (b), which is commonly called the quantum gate teleportation protocol.

In the Choi isomorphism, $|U\rangle := \hat{U} \otimes \hat{I}|\Phi\rangle$ is a Choi vector—Appendix B.2.3—corresponding to the unitary operator \hat{U} —and hence completely characterizes \hat{U} . The quantum gate teleportation protocol uses $|U\rangle$ as a quantum entanglement resource,^{5.10} and it moves the input state $|\Phi\rangle$ on the third qubit to the unitary-transformed state $\hat{U}|\psi\rangle$ on the first qubit. The success probability of the protocol is 1/4. The quantum gate teleportation protocol can be generalized (Problem 5.1) to supermaps for the gate operation and to mixed states for the input state as in the following quantum circuit model



Now consider a state $|\psi\rangle$. In accordance with the quantum gate teleportation

^{5.10}As $\hat{U} \otimes \hat{I}$ only operates *locally*, it does not modify the entanglement characteristics of $|\Phi\rangle$.

protocol in (5.42) and the Choi isomorphism in (5.35), one has

$$\mathcal{F}(|\psi\rangle\langle\psi|)|v_j\rangle = \sum_i |v_i\rangle (\langle v_i| \otimes \langle\Phi|) \left(\hat{C}_{\mathcal{F}} \otimes |\psi\rangle\langle\psi| \right) (|v_j\rangle \otimes |\Phi\rangle) \quad (5.43)$$

Again, as $\hat{C}_{\mathcal{F}}$ is positive if \mathcal{F} is a quantum operation, we use the spectral decomposition (5.39) of $\hat{C}_{\mathcal{F}}$. Finally, define a set of linear operators \hat{F}_μ by

$$\hat{F}_\mu : |\psi\rangle \mapsto \sum_i |v_i\rangle (\langle v_i| \otimes \langle\Phi|) (|\varphi_\mu\rangle \otimes |\psi\rangle). \quad (5.44)$$

Then, we find that

$$\mathcal{F}(|\psi\rangle\langle\psi|) = \sum_\mu \hat{F}_\mu |\psi\rangle\langle\psi| \hat{F}_\mu^\dagger, \quad (5.45)$$

which proves the Kraus representation theorem.

5.1.3 Unitary Representation

A quantum operation can be regarded as a unitary operator on an extended system, which involves an “environment” in addition to the original “system”. Although it is not particularly useful in practical applications, the unitary representation provides a clear physical insight into the underlying physical processes described by the quantum operation.

Suppose that we are given a quantum operation $\mathcal{F} : \mathcal{L}(\mathcal{V}) \rightarrow \mathcal{L}(\mathcal{W})$ represented by the Kraus representation in Eq. (5.2) in terms of the Kraus elements \hat{F}_μ ($\mu = 0, 1, \dots, m - 1$): We want to construct a system-plus-environment model which gives the same effect as \mathcal{F} when the environment is traced out. We need to find a proper vector space \mathcal{E} for the environment and an overall unitary operator \hat{U} acting on the total system $\mathcal{V} \otimes \mathcal{E}$ such that

$$\mathcal{F}(\hat{\rho}) = \text{Tr}_{\mathcal{E}} \hat{U} (\hat{\rho} \otimes |\varepsilon_0\rangle\langle\varepsilon_0|) \hat{U}^\dagger \quad (5.46)$$

for all $\hat{\rho} \in \mathcal{L}(\mathcal{V})$ and a particular state $|\varepsilon_0\rangle \in \mathcal{E}$.^{5.11} We first construct the vector space \mathcal{E} associated with the environment by choosing an orthonormal basis $\{|\varepsilon_\mu\rangle : \mu = 0, \dots, m - 1\}$.^{5.12} We note that the dimension of \mathcal{E} is the same as the number of the Kraus elements \hat{F}_μ in the Kraus representation (5.2) and no larger than $(\dim \mathcal{V}) \times (\dim \mathcal{W})$. Define a unitary operator \hat{U} on $\mathcal{V} \otimes \mathcal{W}$ by requiring that

$$\hat{U} |\psi\rangle \otimes |\varepsilon_0\rangle = \sum_\mu (\hat{F}_\mu |\psi\rangle) \otimes |\varepsilon_\mu\rangle \quad (5.47)$$

^{5.11}The choice of $|\varepsilon_0\rangle$ is completely arbitrary, and one can even choose a mixed state.

^{5.12}Here, just for convenience, we have chosen the basis so as for it to include $|\varepsilon_0\rangle$, but it is not necessary.

for any $|\psi\rangle \in \mathcal{V}$. Clearly, taking the partial trace over the environment reproduces $\mathcal{F}(|\psi\rangle\langle\psi|)$ as one can see from an explicit evaluation

$$\begin{aligned}\text{Tr}_{\mathcal{E}} \hat{U} (|\psi\rangle\langle\psi| \otimes |\varepsilon_0\rangle\langle\varepsilon_0|) \hat{U}^\dagger &= \text{Tr}_{\mathcal{E}} \sum_{\mu\nu} \left(\hat{F}_\mu |\psi\rangle\langle\psi| \hat{F}_\nu^\dagger \right) \otimes |\varepsilon_\mu\rangle\langle\varepsilon_\nu| \\ &= \sum_\mu \hat{F}_\mu |\psi\rangle\langle\psi| \hat{F}_\mu^\dagger \quad (5.48)\end{aligned}$$

This relation is linear and holds for arbitrary vector $|\psi\rangle$, and it should hold for any mixed state $\hat{\rho} = \sum_j |\psi_j\rangle p_j \langle\psi_j|$.

5.1.4 Examples

So far, we have discussed a general description of quantum noisy processes in terms of quantum operations. Let us now take some examples for a single qubit. We will consider some limiting cases that allow us to easily grasp the physical meaning of the Kraus elements.

Phase damping The phase damping process is a decoherence process without involving any relaxation of energy or change in the population over the states. In this sense, it may be regarded as a pure decoherence process without involving any energy relaxation. For this reason, it is also called a *dephasing* process. The unitary representation of a phase damping process in a single-qubit system is given by

$$|0\rangle \otimes |\varepsilon_0\rangle \mapsto |0\rangle \otimes |\varepsilon_0\rangle \quad (5.49a)$$

$$|1\rangle \otimes |\varepsilon_0\rangle \mapsto |1\rangle \otimes |\varepsilon_0\rangle \sqrt{1-p} + |1\rangle \otimes |\varepsilon_1\rangle \sqrt{p}, \quad (5.49b)$$

where $\{|\varepsilon_0\rangle, |\varepsilon_1\rangle\}$ is an orthonormal basis of the vector space \mathcal{E} associated with the environment. It describes that when and only when the system is in $|1\rangle$, the environment changes its state from the initial state $|\varepsilon_0\rangle$ to another orthogonal state $|\varepsilon_1\rangle$ with probability p . Note that the system remains in the same state in both cases. The key point is that nevertheless, the environment “knows” which state the system is in and this knowledge leads to the loss of coherence in the state of the system.

Indeed, the total unitary operator is a controlled- U operator with

$$\hat{U} \doteq \begin{bmatrix} \sqrt{1-p} & -\sqrt{p} \\ \sqrt{p} & \sqrt{1-p} \end{bmatrix} \quad (5.50)$$

in the basis $\{|\varepsilon_0\rangle, |\varepsilon_1\rangle\}$. As a result, when the system is prepared in a coherent superposition, the controlled- \hat{U} operation creates an entanglement between the system and the environment (see Sections 2.2.1 and 2.2.2)

$$(|0\rangle c_0 + |1\rangle c_1) \otimes |\varepsilon_0\rangle \mapsto |0\rangle \otimes |\varepsilon_0\rangle c_0 + |1\rangle \otimes (\hat{U} |\varepsilon_0\rangle) c_1 \quad (5.51)$$

or, more explicitly,

$$(|0\rangle c_0 + |1\rangle c_1) \otimes |\varepsilon_0\rangle \mapsto \left(|0\rangle c_0 + |1\rangle c_1 \sqrt{1-p} \right) \otimes |\varepsilon_0\rangle + |1\rangle \otimes |\varepsilon_1\rangle c_1 \sqrt{p}. \quad (5.52)$$

Due to the entanglement, the final state of the system alone cannot be a pure state (see Section 1.1.2) and coherence in the initial state has been lost through the process.

With the prescription in Section 5.1.1, the Kraus elements are given by

$$\hat{E}_0 \doteq \begin{bmatrix} 1 & 0 \\ 0 & \sqrt{1-p} \end{bmatrix}, \quad \hat{E}_1 \doteq \begin{bmatrix} 0 & 0 \\ 0 & \sqrt{p} \end{bmatrix} \quad (5.53)$$

and the corresponding quantum operation is given by

$$\mathcal{F}(\hat{\rho}) = \hat{E}_0 \hat{\rho} \hat{E}_0^\dagger + \hat{E}_1 \hat{\rho} \hat{E}_1^\dagger \quad (5.54)$$

Note that they are not orthogonal,

$$\text{Tr } \hat{E}_0^\dagger \hat{E}_1 = \sqrt{p(1-p)} \neq 0. \quad (5.55)$$

It is more convenient and efficient to choose mutually orthogonal Kraus elements

$$\hat{F}_0 = \sqrt{\frac{1+\sqrt{1-p}}{2}} \hat{I}, \quad \hat{F}_1 = \sqrt{\frac{1-\sqrt{1-p}}{2}} \hat{S}^z \quad (5.56)$$

In short, the quantum operation for the phase damping process is written as

$$\mathcal{F}(\hat{\rho}) = \frac{1+\sqrt{1-p}}{2} \hat{\rho} + \frac{1-\sqrt{1-p}}{2} \hat{S}^z \hat{\rho} \hat{S}^z \quad (5.57)$$

It is convenient to expand the density operator into

$$\hat{\rho} = \frac{1}{2} \hat{I} + \hat{S}^x \rho_x + \hat{S}^y \rho_y + \hat{S}^z \rho_z, \quad (5.58)$$

where ρ_μ for $\mu = x, y, z$ are real parameters. Then the quantum operation \mathcal{F} gives

$$\mathcal{F}(\hat{\rho}) = \frac{1}{2} \hat{I} + \sqrt{1-p} \hat{S}^x \rho_x + \sqrt{1-p} \hat{S}^y \rho_y + \hat{S}^z \rho_z. \quad (5.59)$$

Note that the coefficient in \hat{S}^z does not change. It asserts that the populations in $|0\rangle$ and $|1\rangle$, $1/2 + \rho_z$ and $1/2 - \rho_z$, do not change. The phase damping process only causes pure dephasing. In particular, at $p \rightarrow 1$, the new density operator approaches,

$$\mathcal{F}(\hat{\rho}) \rightarrow \frac{1}{2} \hat{I} + \hat{S}^z \rho_z. \quad (5.60)$$

The coherence has disappeared completely.

The phase damping process is specified by two Kraus elements.



```
In[7]:= Let[Real, p]
ops = {Sqrt[(1 + Sqrt[1 - p]) / 2], Sqrt[(1 - Sqrt[1 - p]) / 2] * S[3]};
spr = Supermap[ops]

Out[7]= Supermap[{(1 + Sqrt[1 - p])/Sqrt[2], (1 - Sqrt[1 - p])/Sqrt[2]}]
```

Here p is the probability for the phase to be flipped.

```
In[8]:= Let[Real, rho]
rho = 1/2 + p[{1, 2, 3}].S[All]

Out[8]= (1 + Sx ρ1 + Sy ρ2 + Sz ρ3)/2
```

The supermap transforms the above density operator as follows.

```
In[9]:= new = spr[rho]

Out[9]= (1 + Sx ρ1 + Sy ρ2 + Sz ρ3)/2 + Sqrt[1 - p] Sx ρ1 + Sqrt[1 - p] Sy ρ2 + Sqrt[1 - p] Sz ρ3
```

Amplitude damping The amplitude damping process describes the spontaneous decay of the excited state $|1\rangle$ of the system to the ground state $|0\rangle$. The decay is accompanied with the emission of a photon. One can regards that the photon “observes” the system and the information of the system acquired by the photon leads to decoherence. In the unitary representation, the process is described by the overall unitary operator such that

$$|0\rangle \otimes |\varepsilon_0\rangle \mapsto |0\rangle \otimes |\varepsilon_0\rangle , \quad (5.61a)$$

$$|1\rangle \otimes |\varepsilon_0\rangle \mapsto |1\rangle \otimes |\varepsilon_0\rangle \sqrt{1-p} + |0\rangle \otimes |\varepsilon_1\rangle \sqrt{p} . \quad (5.61b)$$

The decay occurs with probability p provided that the system is in the state $|1\rangle$. The Kraus elements are given by

$$\hat{F}_0 \doteq \begin{bmatrix} 1 & 0 \\ 0 & \sqrt{1-p} \end{bmatrix}, \quad \hat{F}_1 \doteq \begin{bmatrix} 0 & \sqrt{p} \\ 0 & 0 \end{bmatrix}. \quad (5.62)$$

They are already orthogonal to each other. The quantum operation describing the amplitude damping process reads as

$$\mathcal{F}(\hat{\rho}) = p\hat{S}^+\hat{\rho}\hat{S}^- + \left(\frac{1 + \sqrt{1-p}}{2} + \frac{1 - \sqrt{1-p}}{2}\hat{S}^z \right) \hat{\rho} \left(\frac{1 + \sqrt{1-p}}{2} + \frac{1 - \sqrt{1-p}}{2}\hat{S}^z \right) \quad (5.63)$$

With the expansion in (5.58), the transformation reads as

$$\mathcal{F}(\hat{\rho}) = \frac{1}{2} + (1-p)\hat{S}^x\rho_x + (1-p)\hat{S}^y\rho_y + \hat{S}^z(1/2 + (1-p)\rho_z). \quad (5.64)$$

In the limit of $p \rightarrow 1$, the new density operator approaches the pure state $|0\rangle$,

$$\mathcal{F}(\hat{\rho}) = \frac{1}{2}\hat{I} + \frac{1}{2}\hat{S}^z = |0\rangle\langle 0|. \quad (5.65)$$

regardless of the initial state. This is due to the relaxation from $|1\rangle$ to $|0\rangle$.

The amplitude damping process is specified by two Kraus elements.

```
In[5]:= Let[Real, p]
ops = {S[10] + Sqrt[1 - p] * S[11], Sqrt[p] * S[4]};
spr = Supermap[ops]
Out[5]= Supermap[{(|0⟩⟨0|)s + √1-p (|1⟩⟨1|)s, √p S+}]
```

Here p is the probability for the phase to be flipped.

```
In[6]:= Let[Real, rho]
rho = 1/2 + p[{1, 2, 3}].S[All]
Out[6]=  $\frac{1}{2} + S^X \rho_1 + S^Y \rho_2 + S^Z \rho_3$ 
```

The supermap transforms the above density operator as follows.

```
In[7]:= new = spr[rho] // Elaborate
Out[7]=  $\frac{1}{2} + \sqrt{1-p} S^X \rho_1 + \sqrt{1-p} S^Y \rho_2 + S^Z \left(p \left(\frac{1}{2} - \rho_3\right) + \rho_3\right)$ 
```

Depolarizing In the depolarizing process, the decoherence occurs symmetrically, and there is no distinction of the specific types of actual decoherence process. The system undergoes an incoherent process with probability p whereas it remains intact with probability $1 - p$. The incoherent process may cause the system to flip the bit value, the phase, or both with equal probability.

The situation can be best described in the unitary representation. Suppose that the system and the environment is in the product state $|\psi\rangle \otimes |\varepsilon_0\rangle$. The decoherence process causes the transition

$$|\psi\rangle \otimes |\varepsilon_0\rangle \mapsto |\psi\rangle \otimes |0\rangle \sqrt{1-p} + (\hat{S}^x |\psi\rangle \otimes |\varepsilon_1\rangle + \hat{S}^y |\psi\rangle \otimes |\varepsilon_2\rangle + \hat{S}^z |\psi\rangle \otimes |\varepsilon_3\rangle) \sqrt{\frac{p}{3}} \quad (5.66)$$

The environment evolves to one of the four mutually orthogonal states. The final states of the environment enables to recognize what process (bit flip, phase flip, or both) has occurred, and hence cause the decoherence on the state of the system.

From the above unitary representation, we can get the Kraus elements

$$\hat{F}_0 = \sqrt{1-p} \hat{I}, \quad \hat{F}_1 = \sqrt{\frac{p}{3}} \hat{S}^x, \quad \hat{F}_2 = \sqrt{\frac{p}{3}} \hat{S}^y, \quad \hat{F}_3 = \sqrt{\frac{p}{3}} \hat{S}^z. \quad (5.67)$$

The Kraus elements are already orthogonal to each other. One can also check that they satisfy the completeness relation

$$\sum_{\mu=0}^3 \hat{F}_\mu^\dagger \hat{F}_\mu = \hat{I} \quad (5.68)$$

as they should. In the Kraus representation with the above Kraus elements, a density operator $\hat{\rho}$ is transformed under the decoherence process as

$$\mathcal{F}(\hat{\rho}) = (1 - p)\hat{\rho} + \frac{p}{3} \left(\hat{S}^x \hat{\rho} \hat{S}^x + \hat{S}^y \hat{\rho} \hat{S}^y + \hat{S}^z \hat{\rho} \hat{S}^z \right). \quad (5.69)$$

In terms of the components, it reads as

$$\mathcal{F}(\hat{\rho}) = \frac{1}{2} \hat{I} + \left(1 - \frac{4p}{3} \right) \left(\hat{S}^x \rho_x + \hat{S}^y \rho_y + \hat{S}^z \rho_z \right). \quad (5.70)$$

It implies that under the process, the “spin” polarization (i.e., the Bloch vector corresponding to the resulting density operator)

$$\mathbf{P} := (\langle \hat{S}^x \rangle, \langle \hat{S}^y \rangle, \langle \hat{S}^z \rangle) \quad (5.71)$$

shrinks by the factor $(1 - 4p/3)$. Hence the name of the process. The state becomes completely random for when $4p/3 = 1$.

The depolarizing process is specified by three Kraus elements.

```
In[1]:= Let[Real, p]
ops = Prepend[Sqrt[p / 3] * S[All], Sqrt[1 - p]];
spr = Supermap[ops]

Out[1]:= Supermap[{Sqrt[1 - p], Sqrt[p / 3] S^x, Sqrt[p / 3] S^y, Sqrt[p / 3] S^z}]
```

Here p is the probability for the phase to be flipped.

```
In[2]:= Let[Real, rho]
rho = 1 / 2 + p[{1, 2, 3}].S[All]

Out[2]:= 1/2 + S^x rho_1 + S^y rho_2 + S^z rho_3
```

The supermap transforms the above density operator as follows.

```
In[3]:= new = spr[rho]
Out[3]:= 1/2 + (1 - 4 p / 3) S^x rho_1 + (1 - 4 p / 3) S^y rho_2 + (1 - 4 p / 3) S^z rho_3
```

5.2 Measurements as Quantum Operations

Generalized measurements (Postulate 3') can be regarded as a special case of quantum operations. Suppose that a measurement is described by a set of measurement operators \hat{M}_m corresponding to measurement outcomes m . The mapping $\mathcal{F}_m \in \mathcal{L}(\mathcal{V})$ defined by

$$\mathcal{F}_m(\hat{\rho}) = \hat{M}_m \hat{\rho} \hat{M}_m^\dagger \quad (5.72)$$

for each m is obviously a quantum operation. This is natural as a measurement process involves the interaction of the system with the measuring devices. Note that the quantum operation \mathcal{F}_m does not preserve the trace in general,

$$0 \leq \text{Tr } \mathcal{F}_m(\rho) \leq 1. \quad (5.73)$$

Physically, $\text{Tr } \mathcal{F}_m(\rho)$ gives the probability to get the outcome m from the measurement process.

The measurement given above is a *selective measurement*. It physically involves separating an ensemble into subensembles that are distinguished by the measurement outcome. [Schwinger \(1959\)](#) conceived a new notion corresponding to the measurement process prior to the stage of selection. It is called a *non-selective measurement*. One can also regard a non-selective measurement as remixing the subensembles after the measurement with the probabilities $\mathcal{F}_m(\hat{\rho})$. A non-selective measurement is thus represented by the quantum operation

$$\mathcal{F}(\hat{\rho}) := \sum_m \mathcal{F}_m(\hat{\rho}) = \sum_m \hat{M}_m \hat{\rho} \hat{M}_m^\dagger. \quad (5.74)$$

In this case, the trace is preserved: $\text{Tr } \mathcal{F}(\hat{\rho}) = 1$ for any $\hat{\rho}$. It follows from the completeness relation, $\sum_m \hat{M}_m^\dagger \hat{M}_m = \hat{I}$, satisfied by the measurement operators.

5.3 Quantum Master Equation

Consider an open quantum system, interacting with its environment. The system is inevitably subject to decoherence processes. Suppose that the system is in $\hat{\rho}(t)$ at time t . To understand the decoherence processes, we want to examine the state $\hat{\rho}(t')$ at later times $t' > t$. The evolution from $\hat{\rho}(t)$ to $\hat{\rho}(t')$ is described by a quantum operation—in this case, a completely positive and trace-preserving superoperator. The operator-sum representation (5.2) guarantees the existence of operators $\hat{F}_\mu(t', t)$ such that

$$\hat{\rho}(t') = \sum_\mu \hat{F}_\mu(t', t) \hat{\rho}(t) \hat{F}_\mu^\dagger(t', t), \quad (5.75a)$$

and satisfying the probability-conserving—trace-preserving—condition

$$\sum_\mu \hat{F}_\mu^\dagger(t', t) \hat{F}_\mu(t', t) = \hat{I} \quad (5.75b)$$

and the orthogonality condition

$$\text{Tr } \hat{F}_\mu^\dagger \hat{F}_\nu = 0 \quad (\mu \neq \nu). \quad (5.75c)$$

However, it turns out that under a specific physical situation, mostly it is difficult to figure out the relevant operators $\hat{F}_\mu(t', t)$ properly describing the given situation.

It may be because the approach attempts to directly determine $\hat{\rho}(t')$ as a function of time t' given the initial condition set by $\hat{\rho}(t)$. It would be more convenient and efficient to express the process in a differential form—a rate equation. After all, both Newton's classical equation of motion and Schrödinger's equation for quantum states are differential equations, describing the rate of changes in the state variables.

Can one express an quantum operation in a set of differential equation that is equivalent to the operator-sum representation? Unfortunately, the answer is “No,” in general. However, under many physically relevant conditions,^{5.13} the operators $\hat{F}_\mu(t', t)$ depend only on the time span $\delta t := t' - t$ but not on the individual instances t' and t . Physically, it implies that the underlying process does not depend on the history, and the assumption is commonly called the *Markov approximation*. Under such conditions, the quantum operation in (5.75) can reformulated in a differential form and the resulting equation,

$$\frac{d\hat{\rho}}{dt} = \mathcal{L}(\hat{\rho}), \quad (5.76)$$

is called the *Lindblad equation* or *quantum master equation*. Here the superoperator \mathcal{L} defined by

$$\mathcal{L}(\hat{\rho}) := -i[\hat{H}, \hat{\rho}] + \sum_\mu \left(\hat{L}_\mu \hat{\rho} \hat{L}_\mu^\dagger - \frac{1}{2} \hat{L}_\mu^\dagger \hat{L}_\mu \hat{\rho} - \frac{1}{2} \hat{\rho} \hat{L}_\mu^\dagger \hat{L}_\mu \right), \quad (5.77)$$

generates the *quantum Markovian dynamics*, and is called the *Lindblad generator*. The Hermitian operator \hat{H} in (5.77) describes the unitary part of the dynamics. For this reason, \hat{H} is often called the *effective Hamiltonian* of the system, but in general it is not the same as the Hamiltonian when the system is isolated. The operators \hat{L}_μ in (5.77) are responsible for non-unitary dynamics and called the *Lindblad operators* or *quantum jump operators*.

It is also customary to rewrite the Lindblad generator (5.77) into the form

$$\frac{d\hat{\rho}}{dt} = -i[\hat{H}, \hat{\rho}] - \{\hat{G}, \hat{\rho}\} + \sum_\mu \hat{L}_\mu \hat{\rho} \hat{L}_\mu^\dagger, \quad (5.77')$$

where

$$\hat{G} := \frac{1}{2} \sum_\mu \hat{L}_\mu^\dagger \hat{L}_\mu. \quad (5.78)$$

Interestingly, ignoring the last term in the quantum jump operators in (5.77'), the solution to the Lindblad equation (5.77') is simply given by

$$\hat{\rho}(t) = e^{-it\hat{H}_{\text{non}}} \hat{\rho}(0) e^{it\hat{H}_{\text{non}}^\dagger}, \quad (5.79)$$

^{5.13}A notable exception is the case where time-dependent external fields are applied on the system.

which resembles the unitary dynamics in (1.32) with the Hamiltonian \hat{H} replaced with the effective *non-Hermitian Hamiltonian*

$$\hat{H}_{\text{non}} := \hat{H} - i\hat{G}. \quad (5.80)$$

The additional term in \hat{G} of the non-Hermitian Hamiltonian makes a significant difference in the evolution governed by Eq. (5.79) as it causes damping—the irreversible population loss in the eigenstates of \hat{H} . In this sense, we call \hat{G} the *effective damping operator*. Although the non-Hermitian Hamiltonian approach does not explain all decoherence processes, it lays out an intuitively appealing picture of open systems and is widely used to describe the effects of finite life time of (effective) energy levels. The non-Hermitian Hamiltonian also provides a good starting point for various more elaborate methods to investigate decoherence processes. A common example is the so-called *quantum jump approach*. It is an approximate method to solve the Lindblad equation combining the non-unitary evolution in (5.79) due to the non-Hermitian Hamiltonian and the “quantum jumps” due to the quantum jump operators \hat{L}_μ (Dum *et al.*, 1992; Plenio & Knight, 1998).

The choice of the Lindblad operators \hat{L}_μ and the effective Hamiltonian \hat{H} is not unique (Breuer & Petruccione, 2002): First, the two sets of Lindblad operators $\{\hat{L}_\mu\}$ and $\{\hat{L}'_\nu\}$ give the same Lindblad equation when

$$\hat{L}'_\nu = \sum_\mu \hat{L}_\mu U_{\mu\nu}, \quad (5.81)$$

where U is a unitary matrix—recall similar unitary freedom for the choice of the Kraus operators in the specification of quantum operations—see Eqs. (5.23) and (5.24)—as well as for the choice of pure states in the specification of mixed states—see Eqs. (1.12) and (1.13). Thanks to the unitary freedom, one can always choose the quantum jump operators to be *mutually orthogonal*,

$$\text{Tr } \hat{L}_\mu^\dagger \hat{L}_\nu = 0 \quad (\mu \neq \nu) \quad (5.82)$$

for all μ and ν . Such a choice is optimal in the sense that $(d^2 - 1)$ quantum jump operators, where $d := \dim \mathcal{V}$, is sufficient for any Lindblad equation. The unitary freedom in (5.81) inherits from the unitary freedom for the choice of the Kraus elements in (5.24). The proof is left for an exercise. Second, the Lindblad generator is also invariant under the inhomogeneous transformations

$$\hat{L}_\mu \rightarrow \hat{L}'_\mu = \hat{L}_\mu + a_\mu, \quad (5.83a)$$

$$\hat{H} \rightarrow \hat{H}' = \hat{H} + \frac{1}{2i} \sum_\mu (a_\mu^* \hat{L}_\mu - a_\mu \hat{L}_\mu^\dagger) + b \quad (5.83b)$$

for any $a_\mu \in \mathbb{C}$ and $b \in \mathbb{R}$. Due to the translational freedom, it is always possible to choose the Lindblad operators to be *traceless*, $\text{Tr } \hat{L}_\mu = 0$. Furthermore, for a given Lindblad equation, it is common to impose the condition $\text{Tr } \hat{H} = 0$ on

the effective Hamiltonian \hat{H} to make it unique. It is straightforward to prove the translational freedom, and again left for an exercise.

As we have pointed out concerning the unitary freedom in the Kraus representation, the unitary freedom does not necessarily imply that one can exploit it to change a given set of Lindblad operators to any arbitrary preferred set of Lindblad operators. This is because the Lindblad operators in (5.81) are not normalized. A notable exception is the two-dimensional case—see Section 5.3.2.

In the remaining of the section, we derive the quantum master equation (5.77) and discuss methods to solve it.

5.3.1 Derivation

It is straightforward to derive the Lindblad equation (5.77) starting from the Kraus representation (5.2) under the Markov assumption—see Breuer & Petruccione (2002) for example. Here we will take a heuristic approach, which is more useful to understand the underlying physics:

As $t' \rightarrow t$ ($\delta t \rightarrow 0$), it is physically required that $\hat{\rho}(t') \rightarrow \hat{\rho}(t)$. It implies that one and only one of $\hat{F}_\mu(\delta t)$ must approach \hat{I} . Let us denote it by $\hat{F}_0(\delta t)$. Up to the first order in δt ,

$$\hat{F}_0(\delta t) \approx \hat{I} + \hat{L}_0 \delta t. \quad (5.84)$$

The rest should vanish $\hat{F}_\mu(\delta t) \rightarrow 0$ for any $\mu > 0$. Since we physically expect that $\hat{\rho}(t') \approx \hat{\rho}(t) + \mathcal{O}(\delta t)$, $\hat{F}_\mu(\delta t)$ must vanish like $\sqrt{\delta t}$ with δt so that $\hat{F}_\mu(\delta t)\hat{\rho}\hat{F}_\mu^\dagger(\delta t) \rightarrow \delta t$. We put

$$\hat{F}_\mu(\delta t) \approx \hat{L}_\mu \sqrt{\delta t} \quad (\mu > 0). \quad (5.85)$$

As \hat{L}_μ ($\mu > 0$) directly proportional to \hat{F}_μ , they are all traceless and mutually orthogonal— $\text{Tr } \hat{L}_\mu^\dagger \hat{L}_\nu = 0$ for $\mu \neq \nu$. The probability conservation condition, Eq. (5.75b), implies that

$$\hat{L}_0 + \hat{L}_0^\dagger = - \sum_{\mu \neq 0} \hat{L}_\mu^\dagger \hat{L}_\mu. \quad (5.86)$$

It suggests that it will be convenient to split \hat{L}_0 into the Hermitian and anti-Hermitian part

$$\hat{L}_0 = -\hat{G} - i\hat{H}, \quad (5.87)$$

where the Hermitian part \hat{G} is fixed by the operators \hat{L}_μ with the relation ($\mu > 0$)

$$\hat{G} = \frac{1}{2} \sum_{\mu=1}^{N^2-1} \hat{L}_\mu^\dagger \hat{L}_\mu. \quad (5.88)$$

whereas \hat{H} remains arbitrary—only determined by \hat{F}_0 , which is linearly independent of \hat{L}_μ ($\mu > 0$). Finally, putting Eqs. (5.84), (5.85), (5.87), and (5.88) into Eq. (5.75) leads to the desired equation (5.77) or, equivalently, to (5.77). Note that in

this particular derivation, the Lindblad operators \hat{L}_μ turn out to be traceless and mutually orthogonal automatically without exploiting the unitary freedom in (5.81)—here the properties inherit from the orthogonality (5.75c) of the Kraus elements.

As mentioned at the beginning of the section, in practice it is difficult to explicitly figure out the quantum operations $\hat{\rho}(t) \mapsto \hat{\rho}(t')$ as a function of time. More common approach is to derive the Lindblad equation, often approximately, by reducing the unitary dynamics of the total system of system plus environment (see also Section 5.1.3). Some examples including perturbative methods are discussed in Breuer & Petruccione (2002).

5.3.2 Examples

Phase damping The Lindblad equation for the phase damping process can be obtained from the Kraus elements in (5.56). We assume that the probability p for the process to occur is proportional to time t , $p = \gamma t$, where γ is the rate of the process per unit time. Expanding the Kraus elements for small t ,

$$\hat{F}_0 \approx \hat{I} - \frac{\gamma}{8} \hat{I}. \quad (5.89)$$

According to (5.87), we identify the effective Hamiltonian and damping operator with

$$\hat{H} = 0, \quad \hat{G} = \frac{\gamma}{8} \hat{I}, \quad (5.90)$$

respectively. Further,

$$\hat{F}_1 \approx \frac{\sqrt{\gamma t}}{2} \hat{S}^z \quad (5.91)$$

implies that there is one Lindblad operator

$$\hat{L}_1 = \frac{\sqrt{\gamma}}{2} \hat{S}^z. \quad (5.92)$$

Overall, the Lindblad generator for the phase damping process is given by

$$\mathcal{L}(\hat{\rho}) = \frac{\gamma}{8} \hat{\rho} + \frac{\gamma}{4} \hat{S}^z \hat{\rho} \hat{S}^z. \quad (5.93)$$

Amplitude damping The Kraus elements for the amplitude damping process are given in (5.62). In the infinitesimal time t ,

$$\hat{F}_0 \approx I - \frac{\gamma}{2} \begin{bmatrix} 0 & 0 \\ 0 & 1 \end{bmatrix}, \quad \hat{F}_1 \approx \sqrt{\gamma} \begin{bmatrix} 0 & 1 \\ 0 & 0 \end{bmatrix} \quad (5.94)$$

Therefore, while the effective Hamiltonian \hat{H} vanishes, the effective damping operator \hat{G} and the Lindblad operator are given by

$$\hat{G} = \frac{\gamma}{4} (1 - \hat{S}^z), \quad \hat{L}_1 = \sqrt{\gamma} \hat{S}^+. \quad (5.95)$$

The Lindblad generator for the amplitude damping is given by

$$\mathcal{L}(\hat{\rho}) = \frac{\gamma}{4} \left\{ (1 - \hat{S}^z) \hat{\rho} + \hat{\rho} (1 - \hat{S}^z) \right\} + \gamma \hat{S}^+ \hat{\rho} \hat{S}^- . \quad (5.96)$$

Depolarizing The Kraus elements for the depolarizing process have been obtained in (5.67). Again, assuming $p = \gamma t$ and expanding the Kraus elements for small t give

$$\hat{F}_0 \approx \hat{I} - \frac{\gamma t}{2} \hat{I}, \quad \hat{F}_\mu = \sqrt{\frac{\gamma t}{3}} \hat{S}^\mu \quad (\mu = x, y, z). \quad (5.97)$$

There are three relevant Lindblad operators

$$\hat{L}_\mu = \sqrt{\frac{\gamma}{3}} \hat{S}^\mu \quad (5.98)$$

for $\mu = 1, 2, 3$, and the effective damping operator is given by

$$\hat{G} = \frac{\gamma}{3} \hat{I}. \quad (5.99)$$

Therefore, the Lindblad generator for the depolarizing process is given by

$$\mathcal{L}(\hat{\rho}) = \frac{\gamma}{2} \hat{\rho} + \frac{\gamma}{3} \sum_{\mu=x,y,z} \hat{S}^\mu \hat{\rho} \hat{S}^\mu . \quad (5.100)$$

General damping For a single qubit, any master equation can be put into the form

$$\frac{d\hat{\rho}}{dt} = -i[\hat{H}, \hat{\rho}] - \{\hat{G}, \hat{\rho}\} + \Gamma_+ \hat{S}^+ \hat{\rho} \hat{S}^- + \Gamma_- \hat{S}^- \hat{\rho} \hat{S}^+ + \Gamma_\phi \hat{S}^z \hat{\rho} \hat{S}^z , \quad (5.101a)$$

where the effective Hamiltonian \hat{H} is arbitrary as long as it is Hermitian, the effective damping operator is given by

$$\hat{G} := \frac{\Gamma_- + \Gamma_+ 2\Gamma_\phi}{4} + \left(\frac{\Gamma_- - \Gamma_+}{4} \right) \hat{S}^z , \quad (5.101b)$$

the real positive parameters Γ_\pm, Γ_ϕ are the rates at which the decoherence processes associated with the quantum jump operators \hat{S}^\pm and \hat{S}^z occur. In this form, the quantum jump operators, \hat{S}^\pm and \hat{S}^z , describe the “simple” transitions—no mixture of different transitions—between the fixed set of states $|0\rangle$ and $|1\rangle$.^{5.14} In general, those states are not the eigenstates of the effective Hamiltonian \hat{H} in the coherent part of the master equation.

^{5.14} \hat{S}^z makes a transition to the same state, but it induces different phase factors depending on the states.

To see that the form in Eq. (5.101) is the most general form of the quantum master equation for a single-qubit system, let us start from the general Lindblad equation

$$\frac{d\hat{\rho}}{dt} = -i[\hat{H}, \hat{\rho}] - \{\hat{G}, \hat{\rho}\} + \sum_{\mu=1}^3 \gamma_{\mu} \hat{A}_{\mu} \hat{\rho} \hat{A}_{\mu}^{\dagger}, \quad (5.102)$$

where

$$\hat{G} := \frac{1}{2} \sum_{\mu} \gamma_{\mu} \hat{A}_{\mu}^{\dagger} \hat{A}_{\mu}. \quad (5.103)$$

The three Lindblad operators are *traceless* and *orthonormal* and $\gamma_{\mu} \geq 0$. Consider another set of orthonormal operators

$$\hat{L}_1 = \hat{S}^+, \quad \hat{L}_2 = \hat{S}^-, \quad \hat{L}_3 = \frac{1}{\sqrt{2}} \hat{S}^z. \quad (5.104)$$

As both sets $\{\hat{A}_{\mu}\}$ and $\{\hat{L}_{\mu}\}$ are orthonormal, there exists a unitary matrix U such that

$$\hat{A}_{\nu} = \sum_{\mu} \hat{L}_{\mu} U_{\mu\nu} \quad (5.105)$$

for all $\nu = 1, 2, 3$. In turn, this implies that there exists a unitary operator $\hat{U} \in \mathcal{L}(\mathcal{V})$ such that

$$\hat{A}_{\nu} = \hat{U} \hat{L}_{\nu} \hat{U}^{\dagger} = \sum_{\mu} \hat{L}_{\mu} U_{\mu\nu} \quad (5.106)$$

for all $\nu = 1, 2, 3$. Putting (5.106) into the Kraus representation (5.102),

$$\hat{U}^{\dagger} \frac{d\hat{\rho}}{dt} \hat{U} = -i \hat{U}^{\dagger} [\hat{H}, \hat{\rho}] \hat{U} - \hat{U}^{\dagger} \{\hat{G}, \hat{\rho}\} \hat{U} + \sum_{\mu} \gamma_{\mu} \hat{L}_{\mu}^{\dagger} \hat{U}^{\dagger} \hat{\rho} \hat{U} \hat{L}_{\mu}^{\dagger}. \quad (5.107)$$

Finally, redefining the operators as^{5.15}

$$\hat{\rho}' := \hat{U}^{\dagger} \hat{\rho} \hat{U}, \quad \hat{H}' := \hat{U}^{\dagger} \hat{\rho} \hat{U}, \quad \hat{G}' := \hat{U}^{\dagger} \hat{G} \hat{U} \quad (5.108)$$

and rescaling the parameters as

$$\Gamma_+ := \gamma_1, \quad \Gamma_- := \gamma_2, \quad \Gamma_{-/-} := \frac{1}{2} \gamma_3 \quad (5.109)$$

one arrives at the Lindblad equation of the form in (5.101) for $\hat{\rho}'$. Since \hat{U} is a unitary transformation, it is nothing but a basis change, and $\hat{\rho}$ and $\hat{\rho}'$ are essentially the same. After solving the Lindblad equation, one can easily get $\hat{\rho}$ by applying the inverse transformation. In this sense, the Lindblad equation in (5.101) is the most general form for a single-qubit system.

^{5.15}Note that \hat{G}' equals to the expression in Eq. (5.101b).

5.3.3 Solution Methods

The Lindblad equation is a linear equation without explicit time dependence, and can always be solved by means of common methods for linear differential equations. More explicitly, in the standard basis, the Lindblad equation can be written as

$$\dot{\rho}_{jk} = \sum_{j'k'} M_{jk;j'k'} \rho_{j'k'} , \quad (5.110)$$

with

$$M_{jk;j'k'} := -i(H_{jj'}\delta_{kk'} - \delta_{jj'}H_{kk'}^*) - (G_{jj'}\delta_{kk'} + \delta_{jj'}G_{kk'}^*) + \sum_{\mu} L_{\mu;jj'}L_{\mu;kk'}^* \quad (5.111)$$

Regarding $\mu := (jk)$ and $\nu := (j'k')$ as collective indices, Eq. (5.110) reads as

$$\dot{\rho}_{\mu} = \sum_{\nu} M_{\mu\nu} \rho_{\nu}, \quad (5.112)$$

which is a typical first-order differential equation for the column vector ρ_{μ} .

Technically, the differential equation (5.112) is not adequate yet to solve because of the conditions that $\rho_{jk} = \rho_{kj}^*$ and that $\sum_j \rho_{jj} = 1$. The latter condition is reflected in the fact that the determinant of the matrix M is always zero. For example, consider a single-qubit system. Suppose that the Lindblad equation is characterized by the effective Hamiltonian

$$\hat{H} = \frac{1}{2}\Omega\hat{S}^z + \frac{1}{2}\Delta\hat{S}^x \quad (5.113)$$

and the Lindblad operators $\sqrt{\Gamma_{\pm}}\hat{S}^{\pm}$. In the matrix form, the Lindblad equation is given by

$$\frac{d}{dt} \begin{bmatrix} \rho_{11} \\ \rho_{12} \\ \rho_{21} \\ \rho_{22} \end{bmatrix} = \begin{bmatrix} -\Gamma_- & \frac{i\Delta}{2} & -\frac{i\Delta}{2} & \Gamma_+ \\ \frac{i\Delta}{2} & -\frac{\Gamma_-}{2} - \frac{\Gamma_+}{2} - i\Omega & 0 & -\frac{i\Delta}{2} \\ -\frac{i\Delta}{2} & 0 & -\frac{\Gamma_-}{2} - \frac{\Gamma_+}{2} + i\Omega & \frac{i\Delta}{2} \\ \Gamma_- & -\frac{i\Delta}{2} & \frac{i\Delta}{2} & -\Gamma_+ \end{bmatrix} \begin{bmatrix} \rho_{11} \\ \rho_{12} \\ \rho_{21} \\ \rho_{22} \end{bmatrix} \quad (5.114)$$

Imposing the conditions, $\rho_{11} + \rho_{22} = 0$ and $\rho_{12} = \rho_{21}^*$, the above equation reads as

$$\frac{d}{dt} \begin{bmatrix} \rho_{11} \\ \text{Re } \rho_{21} \\ \text{Im } \rho_{21} \end{bmatrix} = \begin{bmatrix} -\Gamma & 0 & 0 \\ 0 & -\Gamma/2 & -i\Gamma/2 \\ -i\Delta & -i\Omega & -\Omega \end{bmatrix} \begin{bmatrix} \rho_{11} \\ \text{Re } \rho_{21} \\ \text{Im } \rho_{21} \end{bmatrix} + \begin{bmatrix} \Gamma_+ \\ 0 \\ \Delta/2 \end{bmatrix}, \quad (5.115)$$

where $\Gamma := \Gamma_+ + \Gamma_-$. It is a typical inhomogeneous first-order differential equation and can be solved using the standard methods. If necessary, various numerical methods can also be applied. This method is extensively discussed in Blum (2012).

In the above discussion, the constraints $\hat{\rho}^\dagger = \hat{\rho}$ and $\text{Tr } \hat{\rho} = 1$ have been handled in an ad hoc fashion. They can be dealt with in a systematic way by choosing an appropriate orthonormal basis $\{\hat{B}_\mu : \mu = 0, 1, 2, \dots, n^2 - 1\}$ for $\mathcal{L}(\mathcal{V})$, where n is the dimension of the vector space \mathcal{V} , such that (i) $\hat{B}_0 = \hat{I}/\sqrt{n}$ and (ii) $\hat{B}_\mu^\dagger = \hat{B}_\mu$. Note that the condition (i) implies that the rest elements of the basis are all traceless— $\text{Tr } \hat{B}_\mu = 0$ for $\mu = 1, 2, \dots, n^2 - 1$. We call a basis a *Lindblad basis*. For example, the following operators form a Lindblad basis for a three-level atom:

$$\begin{aligned} & \frac{1}{\sqrt{3}} \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix}, \quad \frac{1}{\sqrt{6}} \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & -2 \end{bmatrix}, \quad \frac{1}{\sqrt{2}} \begin{bmatrix} 1 & 0 & 0 \\ 0 & -1 & 0 \\ 0 & 0 & 0 \end{bmatrix}, \\ & \frac{1}{\sqrt{2}} \begin{bmatrix} 0 & 1 & 0 \\ 1 & 0 & 0 \\ 0 & 0 & 0 \end{bmatrix}, \quad \frac{1}{\sqrt{2}} \begin{bmatrix} 0 & 0 & 1 \\ 0 & 0 & 0 \\ 1 & 0 & 0 \end{bmatrix}, \quad \frac{1}{\sqrt{2}} \begin{bmatrix} 0 & 0 & 0 \\ 0 & 0 & 1 \\ 0 & 1 & 0 \end{bmatrix}, \\ & \frac{1}{\sqrt{2}} \begin{bmatrix} 0 & -i & 0 \\ i & 0 & 0 \\ 0 & 0 & 0 \end{bmatrix}, \quad \frac{1}{\sqrt{2}} \begin{bmatrix} 0 & 0 & -i \\ 0 & 0 & 0 \\ i & 0 & 0 \end{bmatrix}, \quad \frac{1}{\sqrt{2}} \begin{bmatrix} 0 & 0 & 0 \\ 0 & 0 & -i \\ 0 & i & 0 \end{bmatrix}. \end{aligned} \quad (5.116)$$

Let us examine how various quantities are represented in a Lindblad basis: The components of a density operator $\hat{\rho}$ in the expansion

$$\hat{\rho} = \sum_{\mu=0}^{n^2-1} \hat{B}_\mu \rho_\mu \quad (5.117)$$

are given by $\rho_\mu := \text{Tr } \hat{B}_\mu \hat{\rho}$, and they are all real. In particular, $\rho_0 = 1/\sqrt{n}$. More importantly, the Lindblad equation now reads as

$$\dot{\rho}_\mu = \sum_{\nu=1}^{n^2-1} K_{\mu\nu} \rho_\nu + b_\mu \quad (\mu = 1, 2, \dots, n^2 - 1), \quad (5.118)$$

with the generator matrix K given by

$$K_{\mu\nu} := \text{Tr } \hat{B}_\mu^\dagger \mathcal{L}(\hat{B}_\nu), \quad (5.119)$$

where \mathcal{L} is the Lindblad generator in (5.77) or (5.77'), and the inhomogeneous term given by

$$b_\mu := \frac{1}{\sqrt{n}} \text{Tr } \hat{B}_\mu^\dagger \mathcal{L}(\hat{B}_0). \quad (5.120)$$

The inhomogeneous differential equation (5.118) has the solution of the form

$$\rho_\mu(t) = \sum_{\nu=1}^{n^2-1} [e^{Kt}]_{\mu\nu} \rho_\nu(0) + \sum_{\nu} \left[\int_0^t ds e^{Ks} \right]_{\mu\nu} b_\nu. \quad (5.121)$$

Putting this solution for coefficients $\rho_\mu(t)$ back to (5.117) gives the solution $\hat{\rho}(t)$. This method is completely general. Any Lindblad equation can be solved this way, numerically if necessary.

Let us consider a single qubit, and examine a master equation. We assume that the system is initially prepared in a pure state $(|0\rangle + |1\rangle)/\sqrt{2}$.

```
In[1]:= init = (1 + S[1]) / 2;
init // MatrixForm
Out[1]//MatrixForm=

$$\frac{1}{2} \times (1 + S^x)$$

```

This is the effective Hamiltonian.

```
opH = S[3];
```

These are the Lindblad operators.

```
In[2]:= Let[Real, r]
opL = {Sqrt[r["+"]]*S[4], Sqrt[r["-"]]*S[5]}
Out[2]= {S+ \sqrt{\Gamma_+}, S- \sqrt{\Gamma_-}}
```

This is the Lindblad basis we are going to use. It happens to be equivalent to the basis consisting of the Pauli operators.

```
In[3]:= lbs = Elaborate@LindbladBasis[S]
MatrixForm/@(Matrix[#, S] &) /@ lbs
Out[3]= { $\frac{1}{\sqrt{2}}, \frac{S^z}{\sqrt{2}}, \frac{S^x}{\sqrt{2}}, \frac{S^y}{\sqrt{2}}$ }
Out[4]= { $\begin{pmatrix} \frac{1}{\sqrt{2}} & 0 \\ 0 & \frac{1}{\sqrt{2}} \end{pmatrix}, \begin{pmatrix} \frac{1}{\sqrt{2}} & 0 \\ 0 & -\frac{1}{\sqrt{2}} \end{pmatrix}, \begin{pmatrix} 0 & \frac{1}{\sqrt{2}} \\ \frac{1}{\sqrt{2}} & 0 \end{pmatrix}, \begin{pmatrix} 0 & -\frac{i}{\sqrt{2}} \\ \frac{i}{\sqrt{2}} & 0 \end{pmatrix}}$ }
```

Here are the generator matrix K and the inhomogeneous term when the Lindblad equation is written in the standard form of an inhomogeneous first-order differential equation.

```
In[5]:= {mat, vec} = LindbladConvert[opH, opL];
mat // MatrixForm
vec // MatrixForm
Out[5]//MatrixForm=

$$\begin{pmatrix} -\Gamma_- - \Gamma_+ & 0 & 0 & 0 \\ 0 & \frac{1}{2} \times (-2 \dot{i} - \frac{\Gamma_+}{2} - \frac{\Gamma_-}{2}) + \frac{1}{2} \times (2 \dot{i} - \frac{\Gamma_+}{2} - \frac{\Gamma_-}{2}) & -\frac{1}{2} \dot{i} \left(-2 \dot{i} - \frac{\Gamma_+}{2} - \frac{\Gamma_-}{2}\right) + \frac{1}{2} \dot{i} \left(2 \dot{i} - \frac{\Gamma_+}{2} - \frac{\Gamma_-}{2}\right) & \frac{1}{2} \times (-2 \dot{i} - \frac{\Gamma_+}{2} - \frac{\Gamma_-}{2}) + \frac{1}{2} \times (2 \dot{i} - \frac{\Gamma_+}{2} - \frac{\Gamma_-}{2}) \\ 0 & \frac{1}{2} \dot{i} \left(-2 \dot{i} - \frac{\Gamma_+}{2} - \frac{\Gamma_-}{2}\right) - \frac{1}{2} \dot{i} \left(2 \dot{i} - \frac{\Gamma_+}{2} - \frac{\Gamma_-}{2}\right) & \frac{1}{2} \times (-2 \dot{i} - \frac{\Gamma_+}{2} - \frac{\Gamma_-}{2}) + \frac{1}{2} \times (2 \dot{i} - \frac{\Gamma_+}{2} - \frac{\Gamma_-}{2}) & 0 \\ 0 & 0 & 0 & 0 \end{pmatrix}$$

Out[6]//MatrixForm=

$$\begin{pmatrix} -\frac{\Gamma_- + \Gamma_+}{\sqrt{2}} \\ 0 \\ 0 \\ 0 \end{pmatrix}$$

```

This solves the differential equation based on the generator matrix K and the inhomogeneous term.

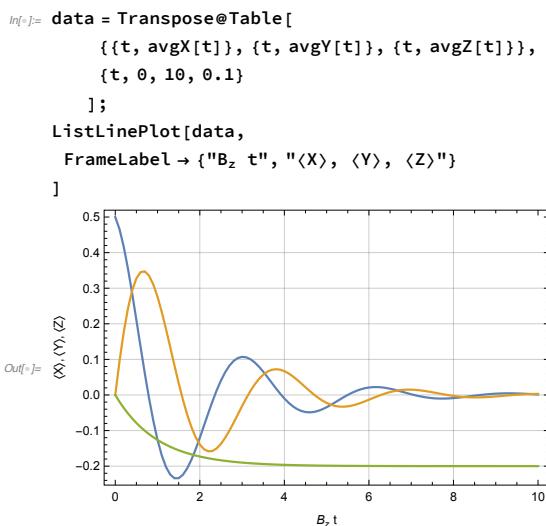
```
In[5]:= Clear[\rho]
ρ[t_] = Block[
{Γ, ρ},
Γ["+"] = .3;
Γ["-"] = .7;
Elaborate@LindbladSolve[opH, opL, init, t] // Chop
];
ρ[t] // MatrixForm
Out[5]/MatrixForm=

$$0.5 + 0.5 e^{-0.5t} \cos[2.t] S^x + (-0.2 + 0.2 e^{-1.t}) S^z + 0.5 e^{-0.5t} S^y \sin[2.t]$$

```

To investigate the physical properties of the solution, we calculate the expectation values of the Pauli operators.

```
In[6]:= {avgX[t_], avgY[t_], avgZ[t_]} = Coefficient[ρ[t], #] & /@ S@{1, 2, 3}
Out[6]= {0.5 e^{-0.5t} \cos[2.t], 0.5 e^{-0.5t} \sin[2.t], -0.2 + 0.2 e^{-1.t}}
```



Consider a three-level atom with the Λ -type level structure.

```
Let[Qudit, A]
```

In the interaction picture, the Hamiltonian looks like this. We have put the two Rabi transition amplitudes to 1.

```
In[8]:= opH = (1/10) A[1 → 1] + (2/10) A[2 → 2] + A[2 → 0] + A[0 → 2] + A[2 → 1] + A[1 → 2];
math = Matrix[opH];
math // MatrixForm
Out[8]/MatrixForm=

$$\begin{pmatrix} 0 & 0 & 1 \\ 0 & \frac{1}{10} & 1 \\ 1 & 1 & \frac{1}{5} \end{pmatrix}$$

```

```
In[1]:= Let[Real, r]
opL = {
  Sqrt[r[0, "-"]]*A[2 → 0],
  Sqrt[r[0, "+"]]*A[0 → 2],
  Sqrt[r[1, "-"]]*A[2 → 1],
  Sqrt[r[1, "+"]]*A[1 → 2]}
matL = Matrix[opL];
MatrixForm@matL

Out[1]= {(|0⟩⟨2|) √{Γ_{0,-}}, (|2⟩⟨0|) √{Γ_{0,+}}, (|1⟩⟨2|) √{Γ_{1,-}}, (|2⟩⟨1|) √{Γ_{1,+}}}

Out[2]= {0 0 √{Γ_{0,-}} 0 0 0 0 0, 0 0 0 0 0 0 0 0, 0 0 √{Γ_{1,-}} 0 0 0 0 0, 0 0 0 0 0 0 0 √{Γ_{1,+}}}

In[3]:= Timing[
ρ[t_] = Block[
  {r, init},
  r[0, "-"] = 0.05;
  r[0, "+"] = 0.01;
  r[1, "-"] = 0.025;
  r[1, "+"] = 0.005;
  init = DiagonalMatrix[{0, 1, 0}];
  LindbladSolve[mathH, matL, init, t]
];
]

Out[3]= {2.56844, Null}

In[4]:= Plot[Evaluate@Diagonal@ρ[t Pi], {t, 0, 10},
  FrameLabel → {"Ω t / π", "Probabilities"},
  PlotRange → All,
  PlotLegends → Automatic]

Out[4]= 

```

There are two drawbacks in the above approach: First, the size of the generator matrix K increases exponentially with the system size, and in many cases, even numerical methods become intractable. In such cases, the quantum jump approach mentioned earlier—Section 5.3—is often used. Second, the resulting solution is given in an explicit matrix representation, and putting the solution in the Kraus representation is tedious or even impractical in many cases. It means that for a physical interpretation of the solution, one needs additional analysis specific to the system or situation.

Interestingly, there is a special class of Lindblad equations that allow for a solution directly in the Kraus representation (Nakazato *et al.*, 2006): Let

$\{|j\rangle : j = 0, \dots, n-1\}$ be the eigenbasis from the effective Hamiltonian \hat{H} so that

$$\hat{H} = \sum_j E_j |j\rangle \langle j|. \quad (5.122)$$

We consider a Lindblad equation of the form

$$\frac{d\hat{\rho}}{dt} = -i[\hat{H}, \hat{\rho}] - \{\hat{G}, \hat{\rho}\} + \sum_{jk} \gamma_{jk} \hat{L}_{jk} \hat{\rho} \hat{L}_{jk}^\dagger, \quad (5.123)$$

where every (normalized) quantum jump operator \hat{L}_{jk} corresponds to an incoherent transition between a pair of eigenstates, $\hat{L}_{jk} := |j\rangle \langle k|$, and γ_{jk} characterizes the rate of the process. Note here that the effective damping operator \hat{G} ,

$$\hat{G} := \frac{1}{2} \sum_{jk} \gamma_{jk} \hat{L}_{jk}^\dagger \hat{L}_{jk} = \frac{1}{2} \sum_k \gamma_k |k\rangle \langle k| \quad (5.124)$$

with

$$\gamma_k := \sum_j \gamma_{jk}, \quad (5.125)$$

commutes with the effective Hamiltonian \hat{H} .

To solve the Lindblad equation (5.123), we first recall that the non-unitary evolution in Eq. (5.79) governed by the non-Hermitian Hamiltonian in Eq. (5.80) corresponds to the solution in the absence of the quantum jump operators ($\gamma_{jk} = 0$). It is therefore natural to define the *generalized interaction picture*

$$\hat{\rho}_I(t) := e^{it\hat{H}_{\text{non}}} \hat{\rho}(t) e^{-it\hat{H}_{\text{non}}^\dagger} \quad (5.126)$$

with respect to the non-Hermitian Hamiltonian $\hat{H}_{\text{non}} = \hat{H} - i\hat{G}$. In this interaction picture, the Lindblad equation reads as

$$\frac{d\hat{\rho}_I}{dt} = \sum_{jk} \gamma_{jk} e^{(\gamma_j - \gamma_k)t} \hat{L}_{jk} \hat{\rho}_I \hat{L}_{jk}^\dagger = \sum_{jk} \gamma_{jk} \hat{R}_{jk}(t), \quad (5.127)$$

where we have defined

$$\hat{R}_{jk}(t) := e^{(\gamma_j - \gamma_k)t} \hat{L}_{jk} \hat{\rho}_I(t) \hat{L}_{jk}^\dagger. \quad (5.128)$$

Since the differential equation (5.127) is equivalent to the integral equation

$$\hat{\rho}_I(t) = \hat{\rho}_I(0) + \sum_{jk} \gamma_{jk} \int_0^t ds \hat{R}_{jk}(s), \quad (5.129)$$

it is now a matter of calculating the new operator $\hat{R}_{jk}(t)$. It follows from (5.127) that the operator $\hat{R}_{jk}(t)$ satisfies the differential equation

$$\frac{d\hat{R}_{jk}}{dt} = \sum_l \Gamma_{kl}^{(j)} \hat{R}_{jl} \quad (5.130)$$

with the matrix $\Gamma^{(j)}$ defined by $\Gamma_{kl}^{(j)} := \delta_{kl}(\gamma_j - \gamma_k) + \gamma_{kl}$. The solution is given by

$$\hat{R}_{jk}(t) = \sum_l \left[e^{t\Gamma^{(j)}} \right]_{kl} \hat{R}_{jl}(0). \quad (5.131)$$

Putting it back into the integral equation (5.129), we finally obtain

$$\hat{\rho}_I(t) = \hat{\rho}(0) + \sum_{jkl} \gamma_{jk} W_{kl}^{(j)}(t) \hat{L}_{jl} \hat{\rho}(0) \hat{L}_{jl}^\dagger \quad (5.132)$$

with

$$W^{(j)}(s) := \int_0^t ds e^{s\Gamma^{(j)}}. \quad (5.133)$$

More explicitly, for the purpose of illustration, the Kraus representation of $\hat{\rho}(t)$ reads as

$$\hat{\rho}(t) = \hat{F}_0(t) \hat{\rho}(0) \hat{F}_0^\dagger(t) + \sum_{jk} \hat{F}_{jk}(t) \hat{\rho}(0) \hat{F}_{jk}^\dagger(t), \quad (5.134a)$$

where the Kraus elements are given by

$$\hat{F}_0(t) := e^{-it\hat{H}_{\text{non}}}, \quad \hat{F}_{jk}(t) := e^{-it\hat{H}_{\text{non}}} |j\rangle \sqrt{\sum_l \gamma_{jl} W_{lk}^{(j)}(t)} \langle k|. \quad (5.134b)$$

5.3.4 Examples Revisited

Here we consider again the limiting cases discussed in Sections 5.1.4 and 5.3.2. Here we will use the method discussed above to solve the Lindblad equations that have been derived in Section 5.3.2. From the solutions, we are supposed to recover the descriptions in terms of quantum operations in Section 5.1.4.

Note that for a single qubit, not surprisingly, the Lindblad basis is given by the Pauli operators. In other words, we expand a density operator $\hat{\rho}$ into

$$\hat{\rho}(t) = \frac{1}{2} \hat{I} + \hat{S}^x \rho_x(t) + \hat{S}^y \rho_y(t) + \hat{S}^z \rho_z(t). \quad (5.135)$$

Phase damping The Lindblad equation in this case is given by [see Eq. (5.93)]

$$\frac{d\hat{\rho}}{dt} = \frac{\gamma}{8} \hat{\rho}(t) + \frac{\gamma}{4} \hat{S}^z \hat{\rho}(t) \hat{S}^z. \quad (5.136)$$

Rewriting the above equation in the Lindblad basis, we have a set of differential equations for the components ρ_μ ($\mu = x, y, z$) of the density operator in Eq. (5.135) as follows

$$\frac{d}{dt} \begin{bmatrix} \rho_x(t) \\ \rho_y(t) \\ \rho_z(t) \end{bmatrix} = -\frac{1}{2}\gamma \begin{bmatrix} 1 & & \\ & 1 & \\ & & 0 \end{bmatrix} \begin{bmatrix} \rho_x(t) \\ \rho_y(t) \\ \rho_z(t) \end{bmatrix}. \quad (5.137)$$

The equation is diagonal and straight forward to solve it to get

$$\rho_x(t) = e^{-\gamma t/2} \rho_x(0), \quad \rho_y(t) = e^{-\gamma t/2} \rho_y(0), \quad \rho_z(t) = \rho_z(0). \quad (5.138)$$

Note that the component $\rho_z(t)$ is constant. Again, this is because the phase damping process does not involves any transition between levels. It only causes pure dephasing.

Consider the phase damping process.

It is governed by a single parameter.

Let[Real, γ]

The Lindblad generator is specified by a single Lindblad operator. Note that in this case, the effective Hamiltonian vanishes.

```
In[7]:= ops = {Sqrt[γ] / 2 × S[3]};
gnr = LindbladGenerator[0, ops]
Out[7]= LindbladGenerator[{0, S^z / 2}]
```

We assume that the system is initially prepared in a pure state.

```
In[8]:= vec = EulerRotation[{0, Pi / 4, 0}, S] ** Ket[];
init = ExpressionFor[Matrix@Dyad[vec, vec], S] // Elaborate;
Out[8]= Cos[π/8] |_-> + |1s> Sin[π/8]
```

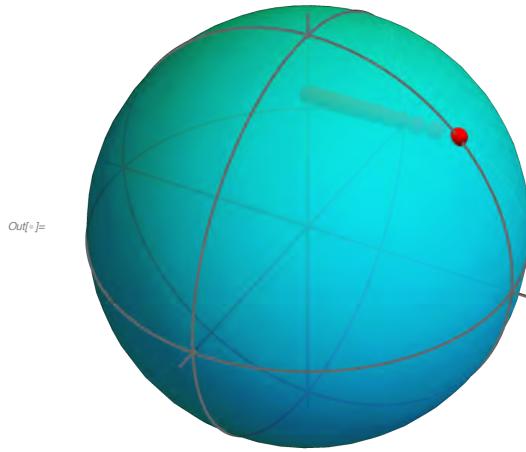
This is the solution of the Lindblad equation.

```
In[9]:= rho[t_] = LindbladSolve[0, ops, init, t];
Out[9]= 1/2 + S^z/(2 √2) + e^-t/2 S^+/(2 √2) + e^-t/2 S^-/(2 √2)
```

As you can see, the time dependence is governed by γt . This indicates that the system has a single time scale, and it is governed by γ . It is thus convenient to use $1/\gamma$ as the unit of time. It is equivalent to put $\gamma=1$.

γ = 1;

```
In[7]:= data = Table[Chop@BlochVector[rho[s Pi]], {s, 0, 3, 0.1}];
BlochSphere[{Red, Bead /@ data}]
```



Amplitude damping Next we consider the amplitude damping process. The Lindblad equation is given by [see (5.96)]

$$\frac{\hat{\rho}}{dt} = \frac{\gamma}{4} \left\{ (1 - \hat{S}^z) \hat{\rho}(t) + \hat{\rho}(t) (1 - \hat{S}^z) \right\} + \gamma \hat{S}^+ \hat{\rho}(t) \hat{S}^- . \quad (5.139)$$

In the Lindblad basis, the above equation reads as

$$\frac{d}{dt} \begin{bmatrix} \rho_x(t) \\ \rho_y(t) \\ \rho_z(t) \end{bmatrix} = -\frac{\gamma}{2} \begin{bmatrix} 1 & & \\ & 1 & \\ & & 2 \end{bmatrix} \begin{bmatrix} \rho_x(t) \\ \rho_y(t) \\ \rho_z(t) \end{bmatrix} + \frac{\gamma}{2} \begin{bmatrix} 0 \\ 0 \\ \sqrt{2} \end{bmatrix} . \quad (5.140)$$

As for the phase damping case, the equation is diagonal. But there is an inhomogeneous term. The solutions are given by

$$\begin{aligned} \rho_x(t) &= e^{-\gamma t/2} \rho_x(0), \\ \rho_y(t) &= e^{-\gamma t/2} \rho_y(0), \\ \rho_z(t) &= e^{-\gamma t/2} \rho_z(0) + \frac{1}{2} (1 - e^{-\gamma t/2}) . \end{aligned} \quad (5.141)$$

We note that

$$\hat{\rho}(t) \rightarrow \frac{1}{2} \hat{I} + \frac{1}{2} \hat{S}^z = |0\rangle \langle 0| \quad (5.142)$$

as $t \rightarrow \infty$. That is, if we wait long enough, the system definitely goes to the pure state $|0\rangle$. This is due to the relaxation from $|1\rangle$ to $|0\rangle$.

Consider the amplitude damping process.

It is governed by a single parameter.

```
Let[Real, γ]
```

The Lindblad generator is specified by a single Lindblad operator. In this case, the effective Hamiltonian vanishes.

```
In[1]:= ops = {Sqrt[γ] * S[4]};
gnr = LindbladGenerator[0, ops]
Out[1]= LindbladGenerator[{0, √γ S+}]
```

We assume that the system is initially prepared in a pure state.

```
In[2]:= vec = EulerRotation[{0, Pi/4, 0}, S] ** Ket[];
init = ExpressionFor[Matrix@Dyad[vec, vec], S] // Elaborate;
Out[2]= Cos[π/8] |_-> + |1s> Sin[π/8]
```

This is the solution of the Lindblad equation.

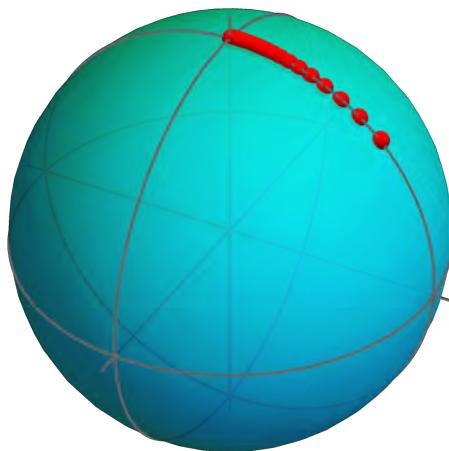
```
In[3]:= rho[t_] = LindbladSolve[0, ops, init, t]
Out[3]= 1/2 + 1/4 e^{-t γ} (-2 + √2 + 2 e^{t γ}) S^z + e^{-t γ/2} S^+ + e^{-t γ/2} S^- / 2 √2
```

As you can see, the time dependence is governed by γt . This indicates that the system has a single time scale, and it is governed by γ . It is thus convenient to use $1/\gamma$ as the unit of time. It is equivalent to put $\gamma=1$.

```
γ = 1;
```

```
In[4]:= data = Table[Chop@BlochVector[rho[s Pi]], {s, 0, 3, 0.1}];
BlochSphere[{Red, Bead @ data}]
```

```
Out[4]=
```



Depolarizing Finally, we examine the depolarizing process. The Lindblad equation is given by [see Eq. (5.100)]

$$\frac{d\hat{\rho}}{dt} = \frac{\gamma}{2}\hat{\rho}(t) + \frac{\gamma}{3}\sum_{\mu=x,y,z}\hat{S}^{\mu}\hat{\rho}(t)\hat{S}^{\mu}. \quad (5.143)$$

The corresponding equation for the components $\rho_{\mu}(t)$ is given by

$$\frac{d}{dt}\begin{bmatrix} \rho_x(t) \\ \rho_y(t) \\ \rho_z(t) \end{bmatrix} = -\frac{4\gamma}{3}\begin{bmatrix} 1 & & \\ & 1 & \\ & & 1 \end{bmatrix}\begin{bmatrix} \rho_x(t) \\ \rho_y(t) \\ \rho_z(t) \end{bmatrix}. \quad (5.144)$$

The solution to the above equation indicates that every component vanishes exponentially

$$\rho_{\mu}(t) = e^{-4\gamma t/3}\rho_{\mu}(0). \quad (5.145)$$

In the long time limit ($t \rightarrow \infty$), the density operator becomes completely random,

$$\hat{\rho}(t) \rightarrow \frac{1}{2}\hat{I}. \quad (5.146)$$

This is the expected characteristic of the depolarizing process.

Consider the depolarizing process.

It is governed by a single parameter.

`Let[Real, γ]`

The Lindblad generator is specified by three Lindblad operators. In this case, the effective Hamiltonian vanishes.

```
In[7]:= ops = Sqrt[γ/3]*S[All];
gnr = LindbladGenerator[0, ops]
Out[7]= LindbladGenerator[{0, Sx/Sqrt[3], Sy/Sqrt[3], Sz/Sqrt[3]}]
```

We assume that the system is initially prepared in a pure state.

```
In[8]:= vec = EulerRotation[{0, Pi/4, 0}, S] ** Ket[];
init = ExpressionFor[Matrix@Dyad[vec, vec], S] // Elaborate;
Out[8]= Cos[π/8]|_-⟩ + |1s⟩ Sin[π/8]
```

This is the solution of the Lindblad equation.

```
In[9]:= rho[t_] = LindbladSolve[0, ops, init, t]
Out[9]= 1/2 + e^-4 t/3 S^z + e^-4 t/3 S^+ + e^-4 t/3 S^-
```

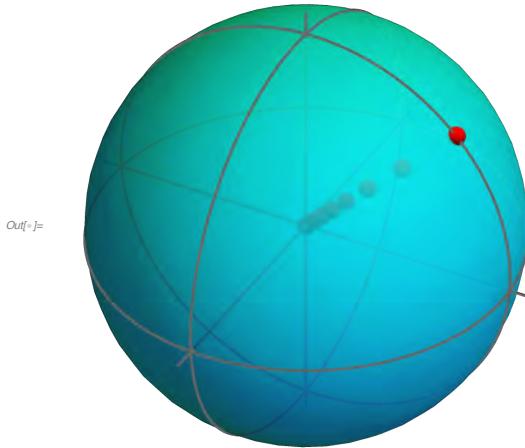
As you can see, the time dependence is governed by γt . This indicates that the system has a single time scale, and it is governed by γ . It is thus convenient to use $1/\gamma$ as the unit of time. It is equivalent to put $\gamma=1$.

```

y = 1;

In[1]:= data = Table[Chop@BlochVector[rho[s Pi]], {s, 0, 3, 0.1}];
BlochSphere[{Red, Bead /. data}]

```



Problems

- 5.1. By explicitly evaluating the quantum circuit model in (5.42), prove the quantum gate teleportation protocol.
- 5.2. Consider a quantum register of two qubits. Let \mathcal{F} be a quantum operation on the quantum register, specified by the Kraus elements

$$\begin{aligned}\hat{F}_0 &= \sqrt{1 - p_1 - p_2 - p_3} \hat{I}, \\ \hat{F}_1 &= \sqrt{p_1} \hat{S}_1^-, \quad \hat{F}_2 = \sqrt{p_2} \hat{S}_2^-, \quad \hat{F}_3 = \sqrt{p_3} \hat{S}_1^- \hat{S}_2^-.\end{aligned}\quad (5.147)$$

- (a) Construct a quantum circuit model to generate the Choi operator $\hat{C}_{\mathcal{F}}$ associated with the quantum operation \mathcal{F} . Note that the input state of the system (including an auxiliary quantum register if necessary) must be $|0\rangle \equiv |0\rangle \otimes |0\rangle \otimes \dots$.

Hint: See the quantum circuit model representation in (5.35) to generate the Choi operator. You have to generate the maximally entangled state starting from $|0\rangle$.

- (b) Write down the Choi operator $\hat{C}_{\mathcal{F}}$ in terms of the Pauli operators \hat{S}_1^μ and \hat{S}_2^ν on the two qubits.

- 5.3. Consider the following Lindblad equation for a single qubit:

$$\frac{d\hat{\rho}}{dt} = -i[\hat{H}, \hat{\rho}] - i\{\hat{G}, \hat{\rho}\} + \gamma_- |0\rangle \langle 1| \hat{\rho} |1\rangle \langle 0| + \gamma_1 |1\rangle \langle 1| \hat{\rho} |1\rangle \langle 1|,\quad (5.148)$$

where the effective Hamiltonian \hat{H} and the damping operator \hat{G} are given by

$$\hat{H} = E_0 |0\rangle\langle 0| + E_1 |1\rangle\langle 1|, \quad \hat{G} = \frac{1}{2}(\gamma_- + \gamma_\phi) |1\rangle\langle 1|. \quad (5.149)$$

Put $E_0 = 0$ without loss of generality and assume $0 < E_1$. γ_- and γ_ϕ are responsible for the amplitude and phase damping process, respectively. Solve the Lindblad equation and represent the solution $\hat{\rho}(t)$ in the Kraus representation form.

- 5.4. Consider a three-level atom. Suppose that it is subject to an interaction to its environment, which is described by the following Lindblad equation

$$\frac{d\hat{\rho}}{dt} = -i[\hat{H}, \hat{\rho}] - i\{\hat{G}, \hat{\rho}\} + \eta_0 |0\rangle\langle 2| \hat{\rho} |2\rangle\langle 0| + \eta_1 |1\rangle\langle 2| \hat{\rho} |2\rangle\langle 1|, \quad (5.150)$$

where $|0\rangle$ and $|1\rangle$ are the ground-state levels and $|2\rangle$ is the excited state. The effective Hamiltonian is given by

$$\hat{H} = E_1 |1\rangle\langle 1| + E_2 |2\rangle\langle 2| \quad (0 \leq E_1 \ll E_2), \quad (5.151)$$

where we have set the ground-state energy to zero $E_0 = 0$. The damping operator is given by

$$\hat{G} = \frac{1}{2}(\gamma_0 + \gamma_1) |2\rangle\langle 2|. \quad (5.152)$$

Solve the Lindblad equation and put the solution $\hat{\rho}(t)$ in the Kraus representation form. Assume that the atom is initially prepared in the pure state $|1\rangle$.

Chapter 6

Quantum Error-Correction Codes

- August 4, 2021 (v1.11)

Nothing is perfect and everything is prone to errors. But what makes quantum information different from classical information when it comes to error correction?

Any physical system inevitably interacts with its surroundings—the environment. The effect is particularly severe to quantum systems. Quantum information is a delicate state of superposition. The environment can knock the system out of the delicate superposition. It leads to decoherence and causes damage to quantum information. Furthermore, quantum gates involved in quantum information processing reside in a continuum of unitary transformations. Implementation of quantum gates with perfect accuracy is unrealistic. Even worse, small imperfections may accumulate and result in serious errors in the state undergoing the gate operations. On these accounts, the characteristic of errors in quantum information is clear: quantum errors are continuous. Detection of continuous errors, not to mention correction of them, already seems formidable.

The principles of quantum mechanics themselves make handling quantum errors challenging. In classical information processing, a basic approach is to make duplicate copies before information processing and compare the outputs of different copies to check for any error. For quantum information, this approach is not allowed because the no-cloning theorem (see Section 1.3.1) prevents copying unknown quantum states. Measurement puts another obstacle. Classically, one can probe the system and, if necessary, correct the error. In quantum mechanics, this tactic does not work as measurement disturbs quantum states.

In this chapter, we will see that amazingly, despite the apparent difficulties, it is possible to correct quantum errors successfully. It is achieved by suitably encoding quantum information. If quantum information is encoded appropriately, then it can be recovered by correcting merely a discrete set of errors as long as the

error rate is not too high. We will find the conditions for quantum error-correction codes to protect successfully against probable errors.

6.1 Elementary Examples: Nine-Qubit Code

We start with some elementary examples. They require relatively large resources for given types of errors, and in this sense, there are more efficient methods. However, their simple structure makes it easier to grasp the fundamental ideas behind quantum error-correction codes. They exemplify how to encode quantum information to protect against errors, how to detect error syndrome, and how to recover quantum information from corrupted data.

6.1.1 Bit-Flip Errors

Suppose that the quantum computer is subject to quantum noise which flips the basis states of the individual qubits, $|0\rangle \leftrightarrow |1\rangle$, with probability p . The error is called the *bit-flip error*, and we are to examine a code to correct it.

To protect an arbitrary single-qubit state $|\psi\rangle = |0\rangle c_0 + |1\rangle c_1$ against the bit-flip error, one has to encode it to multiple qubits. For example, one encodes $|\psi\rangle$ to the three-qubit state $|\bar{\psi}\rangle = |000\rangle c_0 + |111\rangle c_1$. In this encoding, the logical basis states are

$$|\bar{0}\rangle := |000\rangle, \quad |\bar{1}\rangle := |111\rangle, \quad (6.1)$$

and distinguished from the physical basis states $|0\rangle$ and $|1\rangle$ of each qubit. Therefore, the subspace spanned by the two logical basis states is the *code space*. The projection onto the code space is given by $\hat{P}_0 := |000\rangle\langle 000| + |111\rangle\langle 111|$. The encoding procedure can be achieved through a quantum circuit model (recall Fig. 2.5)



or (recall Problem 2.5)



To see how this encoding protects against bit-flip error, suppose that the error occurs on the first qubit—the error is thus described by the operator $\hat{X} \otimes \hat{I} \otimes \hat{I}$. Then, the encoded state changes to $|\bar{\psi}'\rangle = |100\rangle c_0 + |011\rangle c_1$. As this state is orthogonal to the original state $|\bar{\psi}\rangle$, in principle it can be discriminated by a proper measurement. Indeed, the measurement associated with the projection $\hat{P}_1 := |100\rangle\langle 100| + |011\rangle\langle 011|$ onto the subspace spanned by $|100\rangle$ and $|011\rangle$ can

do it. Similarly, bit flips on the second and third qubit modify the encoded state into $|\bar{\psi}''\rangle = |010\rangle c_0 + |101\rangle c_1$ and $|\bar{\psi}'''\rangle = |001\rangle c_0 + |110\rangle c_1$, respectively. These states reside in the subspaces corresponding to the projection operators $\hat{P}_2 := |010\rangle \langle 010| + |101\rangle \langle 101|$ and $\hat{P}_3 := |001\rangle \langle 001| + |110\rangle \langle 110|$, respectively. As the bit-flip errors on individual qubits bring the encoded state $|\psi\rangle$ to mutually orthogonal subspaces, or equivalently, $\hat{P}_i \hat{P}_j = \delta_{ij} \hat{P}_j$, one can set up a projection measurement that is described by the projectors $\hat{P}_0, \hat{P}_1, \hat{P}_2, \hat{P}_3$ to tell on which qubit the error has actually occurred (if at all). The measurement result is called the *error syndrome*, and the measurement is called the *error-detection* procedure.

It is interesting to note that the states $|\bar{\psi}\rangle, |\bar{\psi}'\rangle, |\bar{\psi}''\rangle$, and $|\bar{\psi}'''\rangle$ are all simultaneous eigenstates of the two observables

$$\hat{M}_1 := \hat{Z} \otimes \hat{Z} \otimes \hat{I}, \quad \hat{M}_2 := \hat{I} \otimes \hat{Z} \otimes \hat{Z}. \quad (6.3)$$

Further, since they are mutually orthogonal, they must belong to different pairs of eigenvalues of \hat{M}_1 and \hat{M}_2 . Indeed, we see by inspection that they belong to the pairs of eigenvalues $(1, 1), (-1, 1), (-1, -1)$ and $(1, -1)$, respectively. Therefore, the measurements of \hat{M}_1 and \hat{M}_2 give exactly the same information about the error syndrome as the projection measurement associated with \hat{P}_j outlined above. Less obvious at this stage, this method of error-detection is easier to extend—and more closely related to the stabilizer formalism to be discussed in Sections 6.3 and 6.4—than the above measurement in terms of the projectors \hat{P}_j , and we will mainly focus on this method from now on.

It is crucial for the error-detection procedure does not reveal the details of the encoding, i.e., the amplitudes c_0 and c_1 in $|\bar{\psi}\rangle$; otherwise, the measurement of error syndrome would destroy the superposition in quantum states. For example, the measurement of \hat{M}_1 just compares the values of the first and second qubit, giving $+1$ if the values are the same and -1 if they are different, but never reveals the individual values. Similarly, measuring \hat{M}_2 only compares the values of the second and third qubit, but does not disclose the individual values. Nevertheless, it is still possible to tell where the error occurs. For example, suppose that the quantum noise flips the first qubit. Comparing the values of the first two qubits tells that the error has occurred on either of the first qubits, but not exact which. Comparing the second and third qubit tells that the error has occurred neither (or both) of the qubits. Assuming that the error rate is sufficiently low, the two comparisons can predict successfully that the error occurred on the first qubit.

Once the error-detection procedure identifies the error syndrome, the next step is to correct the error and recover the original encoded state $|\psi\rangle$. This step is called the *recovery* procedure. For bit-flip error, the recovery procedure is quite simple. Depending on the qubit that suffers from the error, one has just to apply the gate operation $\hat{X} \otimes \hat{I} \otimes \hat{I}$, $\hat{I} \otimes \hat{X} \otimes \hat{I}$, or $\hat{I} \otimes \hat{I} \otimes \hat{X}$; or, of course, nothing if the measurement indicates that no error has occurred.

Following the error-correction procedure above, the original state is recovered successfully as long as either no error occurs at all or the bit-flip error occurs on

only one of the three qubits. The probability for the successful recovery is thus given by $(1 - p)^2 + 3p(1 - p)^2 = 1 - 3p^2 + 2p^2$. Compared with the probability $1 - p$ for a bare—i.e., non-encoded—state to remain uncorrupted, the encoding and subsequent error-correction procedure improve the reliability of storage of the quantum state as long as $p < 1/2$. This rather crude estimation of the quality of the quantum error-correction procedure—a more complete analysis is discussed later—suggests a first impression how the quantum error-correction code works.

Suppose that we want to encode a single-qubit state to protect it against the bit-flip error.

```
In[1]:= vec = ProductState[S[1] → {c[0], c[1]}, "Label" → Ket[ψ]]
Out[1]= (c₀ | 0⟩ + c₁ | 1⟩)ₜ₁
```

The encoding is achieved by a quantum circuit model of the form.

```
In[2]:= qc = QuantumCircuit[vec,
  LogicalForm[Ket[], S@{2, 3}], CNOT[S[1], S[2]], CNOT[S[1], S[3]]]
Out[2]=
```

```
In[3]:= out = DefaultForm@ExpressionFor[qc];
out // LogicalForm
Out[3]= c₀ | 0ₛ₁ 0ₛ₂ 0ₛ₃⟩ + c₁ | 1ₛ₁ 1ₛ₂ 1ₛ₃⟩
```

Here is another quantum circuit model giving the same encoding.

```
In[4]:= qc2 = QuantumCircuit[vec,
  LogicalForm[Ket[], S@{2, 3}], CNOT[S[1], S[2]], CNOT[S[2], S[3]]]
Out[4]=
```

```
In[5]:= out2 = DefaultForm@ExpressionFor[qc2];
out2 // LogicalForm
Out[5]= c₀ | 0ₛ₁ 0ₛ₂ 0ₛ₃⟩ + c₁ | 1ₛ₁ 1ₛ₂ 1ₛ₃⟩
```

Consider a bit-flip error occurs randomly

```
In[6]:= errors = Join[{1}, S@{1, 2, 3}, 1];
opE = RandomChoice[errors];
new = opE ** out;
new // LogicalForm
Out[6]= {1, S₁¹, S₂¹, S₃¹}
Out[7]= S₁¹
Out[8]= c₁ | 0ₛ₁ 1ₛ₂ 1ₛ₃⟩ + c₀ | 1ₛ₁ 0ₛ₂ 0ₛ₃⟩
```

This gives the error syndrome.

```
In[=] := {(S[1, 3] ** S[2, 3] ** new) / new,
           (S[2, 3] ** S[3, 3] ** new) / new} // Simplify
Out[=] := {-1, 1}
```

]]

6.1.2 Phase-Flip Error

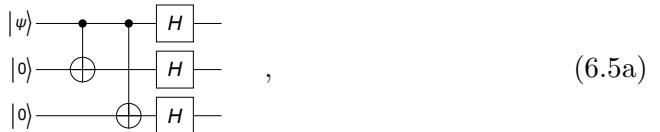
As already mentioned, a quantum state is a superposition of the logical basis state, and any modulation of superposition coefficients without altering the logical basis states results in an error. In particular, a mere change in relative phase in the coefficients leads to a deviation from the original state although it does not change the occupation probabilities in the logical basis states. Let us consider an extreme case where the relative phase of a single qubit is shifted by π —the relative sign of the superposition coefficients is flipped from $+1$ to -1 and vice versa—with probability p . It is called the *phase-flip error*, and the corresponding error operator is \hat{Z} . Such an error is particularly interesting because it has no classical counterpart—classically, the phase does not play any role.

Nevertheless, the phase-flip error is essentially no different from the bit-flip error. This gets clear immediately when one observes that the phase-flip error switches the states $|\pm\rangle := (|0\rangle \pm |1\rangle)/\sqrt{2}$ to each other. Therefore, a change of basis from $\{|0\rangle, |1\rangle\}$ to $\{|+\rangle, |-\rangle\}$ just converts the phase-flip error to the bit-flip error, which has already been addressed above. Apart from the proper choice of basis, the whole error-correction procedure including encoding, error-detection, and recovery is the same as protecting against bit-flip errors. It looks a bit ironic that the phase-flip error, which appears genuinely-quantum, can be countered by a basis change, another feature of quantum mechanics with no classical equivalent. The fact suggests that all types of errors can be handled on an equal footing as the bit-flip error, and inspires ideas about how to protect against quantum noise.

Although already clear from the above observation, for later use we construct the procedure to protect against phase-flip errors. The encoding chooses

$$|\bar{0}\rangle := |+++ \rangle, \quad |\bar{1}\rangle := |--- \rangle \quad (6.4)$$

as the logical basis states. The unitary transformation implementing the encoding can be described a quantum circuit model of the form



or



Compared with the corresponding quantum circuit models in (6.2), the above quantum circuit models have additional Hadamard gates, $\hat{H}^{\otimes 3}$, to account for the basis change mentioned above. Phase-flip errors on one or fewer qubits of the three can be detected by measuring the observables [recall Eq. (6.3)]

$$\hat{M}'_1 = \hat{H}^{\otimes 3} \hat{M}_1 \hat{H}^{\otimes 3} = \hat{X} \otimes \hat{X} \otimes \hat{I}, \quad (6.6a)$$

$$\hat{M}'_2 = \hat{H}^{\otimes 3} \hat{M}_2 \hat{H}^{\otimes 3} = \hat{I} \otimes \hat{X} \otimes \hat{X}, \quad (6.6b)$$

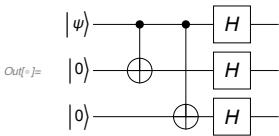
as $\hat{H}\hat{Z}\hat{H} = \hat{X}$. That is, the phase-flip error occurs on the first, second and third qubit leads to the measurement results $(-1, 1)$, $(-1, -1)$, and $(1, -1)$, respectively, and the outcome will be $(1, 1)$ if no error occurs. Once the error syndrome is diagnosed, the error can be fixed by applying the recovery operator $\hat{I} \otimes \hat{I} \otimes \hat{I}$ (doing nothing), $\hat{Z} \otimes \hat{I} \otimes \hat{I}$, $\hat{I} \otimes \hat{Z} \otimes \hat{I}$, or $\hat{I} \otimes \hat{I} \otimes \hat{Z}$ as $\hat{H}\hat{X}\hat{H} = \hat{Z}$.

Suppose that we want to encode a single-qubit state to protect it against the phase-flip error.

```
In[7]:= vec = ProductState[S[1] → {c[0], c[1]}, "Label" → Ket[ψ]]
Out[7]= (c₀ | 0⟩ + c₁ | 1⟩)ₜ₁
```

The encoding is achieved by a quantum circuit model of the form.

```
In[8]:= qc = QuantumCircuit[vec, LogicalForm[Ket[], S@{2, 3}],
  CNOT[S[1], S[2]], CNOT[S[1], S[3]], S[{1, 2, 3}, 6]]
```



```
In[9]:= out = DefaultForm@ExpressionFor[qc];
out // LogicalForm
Out[9]= (c₀ + c₁) | 0s₁ 0s₂ 0s₃⟩ + (c₀ - c₁) | 0s₁ 0s₂ 1s₃⟩ + (c₀ - c₁) | 0s₁ 1s₂ 0s₃⟩ + (c₀ + c₁) | 0s₁ 1s₂ 1s₃⟩ +
          2 √2                               2 √2                               2 √2                               2 √2 +
(c₀ - c₁) | 1s₁ 0s₂ 0s₃⟩ + (c₀ + c₁) | 1s₁ 0s₂ 1s₃⟩ + (c₀ + c₁) | 1s₁ 1s₂ 0s₃⟩ + (c₀ - c₁) | 1s₁ 1s₂ 1s₃⟩
          2 √2                               2 √2                               2 √2                               2 √2
```

To compare the above result with the desired state, let us construct the encoding explicitly.

```
In[10]:= new = ProductState[S@{1, 2, 3} → {1, 1} / Sqrt[2]] × c[0] +
  ProductState[S@{1, 2, 3} → {1, -1} / Sqrt[2]] × c[1]
Out[10]= c₁ (| 0⟩ / √2 - | 1⟩ / √2)ₜ₁ ⊗ (| 0⟩ / √2 - | 1⟩ / √2)ₜ₂ ⊗ (| 0⟩ / √2 - | 1⟩ / √2)ₜ₃ +
  c₀ (| 0⟩ / √2 + | 1⟩ / √2)ₜ₁ ⊗ (| 0⟩ / √2 + | 1⟩ / √2)ₜ₂ ⊗ (| 0⟩ / √2 + | 1⟩ / √2)ₜ₃
```

```
In[11]:= out - new // Elaborate
```

```
Out[11]= 0
```

Here is another quantum circuit model giving the same encoding.

```
In[=]:= qc2 = QuantumCircuit[in, LogicalForm[Ket[], S@{2, 3}],  
CNOT[S[1], S[2]], CNOT[S[2], S[3]], S[{1, 2, 3}, 6]]
```

```
Out[=]=
```

$$\frac{(c_0 + c_1) \left| 0_{S_1} 0_{S_2} 0_{S_3} \right\rangle}{2\sqrt{2}} + \frac{(c_0 - c_1) \left| 0_{S_1} 0_{S_2} 1_{S_3} \right\rangle}{2\sqrt{2}} + \frac{(c_0 - c_1) \left| 0_{S_1} 1_{S_2} 0_{S_3} \right\rangle}{2\sqrt{2}} + \frac{(c_0 + c_1) \left| 0_{S_1} 1_{S_2} 1_{S_3} \right\rangle}{2\sqrt{2}} +$$

$$\frac{(c_0 - c_1) \left| 1_{S_1} 0_{S_2} 0_{S_3} \right\rangle}{2\sqrt{2}} + \frac{(c_0 + c_1) \left| 1_{S_1} 0_{S_2} 1_{S_3} \right\rangle}{2\sqrt{2}} + \frac{(c_0 + c_1) \left| 1_{S_1} 1_{S_2} 0_{S_3} \right\rangle}{2\sqrt{2}} + \frac{(c_0 - c_1) \left| 1_{S_1} 1_{S_2} 1_{S_3} \right\rangle}{2\sqrt{2}}$$

6.1.3 Shor's Nine-Qubit Code

So far we have considered specific types of errors, the bit-flip and phase-flip errors. Obviously, realistic errors are far more general, not only the combination of the above two but also any continuous change of the complex amplitudes in superposition of quantum states. It may sound rather surprising at a first glance, but it turns out that roughly speaking, if a quantum error-correction code can correct the bit-flip and phase-flip errors, then it can correct any arbitrary errors. Before we make the statement and the corresponding construction of quantum error-correction codes in subsequently sections, here we provide an example, Shor's 9-qubit code, that just builds up on the bit-flip and phase-flip correction codes discussed above, and yet corrects completely arbitrary single-qubit errors.

According to the discussion above, protecting against the bit-flip and phase-flip error each requires three qubits to encode physical quantum states. Hence, the encoding in Shor's code needs 9 qubits to protect against both types of errors and their combinations. More specifically, following the phase-flip correction code, we first encode the physical basis states $|0\rangle$ and $|1\rangle$ in the three-qubit logical basis states $|+++ \rangle$ and $|--- \rangle$, and then encode each single-qubit state on three qubits using the bit-flip encoding scheme, that is, $|+\rangle$ by $(|000\rangle + |111\rangle)/\sqrt{2}$ and $|-\rangle$ by $(|000\rangle - |111\rangle)/\sqrt{2}$. One can regard that the phase-flip correction code is encoded on three blocks, and the bit-flip correction code on each block is encoded on the three qubits. The code words of the overall encoding are given by

$$|\bar{0}\rangle := \frac{(|000\rangle + |111\rangle) \otimes (|000\rangle + |111\rangle) \otimes (|000\rangle + |111\rangle)}{\sqrt{8}}, \quad (6.7a)$$

$$|\bar{1}\rangle := \frac{(|000\rangle - |111\rangle) \otimes (|000\rangle - |111\rangle) \otimes (|000\rangle - |111\rangle)}{\sqrt{8}}. \quad (6.7b)$$

The encoding process can be implemented either—or equivalent variants—of the quantum circuit models in Fig. 6.1. The first part, as separated from the latter part by the vertical dotted line, of each quantum circuit model is for the phase-flip

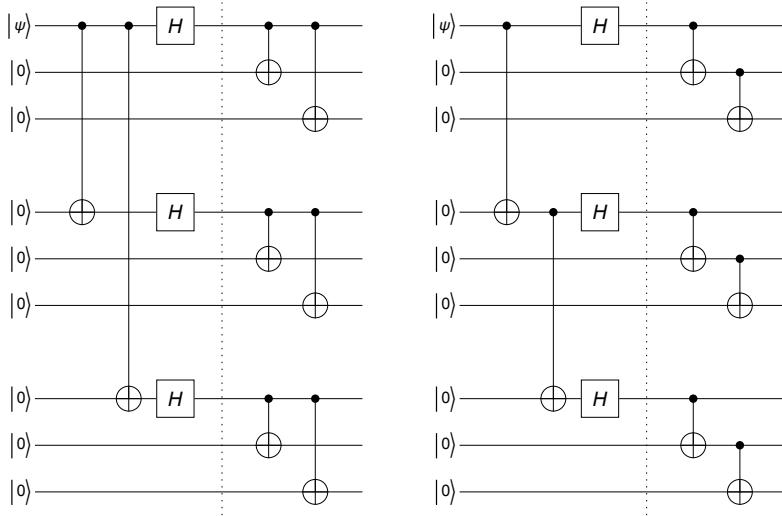


Figure 6.1: Two versions of quantum circuit model for the encoding process in Shor’s 9-qubit quantum error-correction code.

encoding—recall the quantum circuit models in (6.2)—and the second part is to encode each block based on the bit-flip encoding—recall the quantum circuit models in (6.5).

The syndrome of bit-flip error on the first block can be diagnosed by measuring the following two observables

$$(\hat{Z} \otimes \hat{Z} \otimes \hat{I}) \otimes (\hat{I} \otimes \hat{I} \otimes \hat{I}) \otimes (\hat{I} \otimes \hat{I} \otimes \hat{I}), \quad (6.8a)$$

$$(\hat{I} \otimes \hat{Z} \otimes \hat{Z}) \otimes (\hat{I} \otimes \hat{I} \otimes \hat{I}) \otimes (\hat{I} \otimes \hat{I} \otimes \hat{I}). \quad (6.8b)$$

This is the way the bit-flip error is detected for a state encoded in three qubits. Similarly, the bit-flip errors on the rest blocks are diagnosed by measurement of the observables

$$(\hat{I} \otimes \hat{I} \otimes \hat{I}) \otimes (\hat{Z} \otimes \hat{Z} \otimes \hat{I}) \otimes (\hat{I} \otimes \hat{I} \otimes \hat{I}), \quad (6.8c)$$

$$(\hat{I} \otimes \hat{I} \otimes \hat{I}) \otimes (\hat{I} \otimes \hat{Z} \otimes \hat{Z}) \otimes (\hat{I} \otimes \hat{I} \otimes \hat{I}), \quad (6.8d)$$

$$(\hat{I} \otimes \hat{I} \otimes \hat{I}) \otimes (\hat{I} \otimes \hat{I} \otimes \hat{I}) \otimes (\hat{Z} \otimes \hat{Z} \otimes \hat{I}), \quad (6.8e)$$

$$(\hat{I} \otimes \hat{I} \otimes \hat{I}) \otimes (\hat{I} \otimes \hat{I} \otimes \hat{I}) \otimes (\hat{I} \otimes \hat{Z} \otimes \hat{Z}). \quad (6.8f)$$

Once the qubit where the bit-flip error has occurred is identified, the error can be corrected by applying \hat{X} on the qubit.

The phase-slip error on any single qubit switches the relative sign in the encoded state of the whole block that it belongs to. Following the phase-flip correction

code, it is thus sufficient to compare the signs of different blocks. The observable to compare the signs of the first two blocks is

$$(\hat{X} \otimes \hat{X} \otimes \hat{X}) \otimes (\hat{X} \otimes \hat{X} \otimes \hat{X}) \otimes (\hat{I} \otimes \hat{I} \otimes \hat{I}). \quad (6.9a)$$

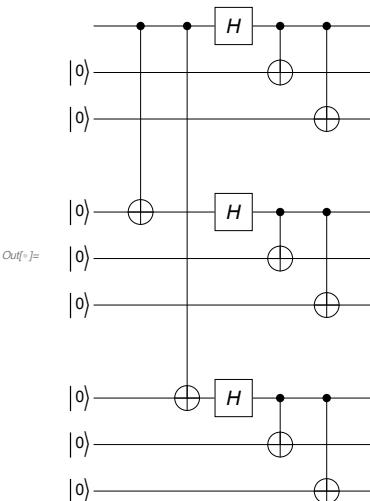
Similarly, the signs of the second and third blocks can be compared through the measurement of the observable

$$(\hat{I} \otimes \hat{I} \otimes \hat{I}) \otimes (\hat{X} \otimes \hat{X} \otimes \hat{X}) \otimes (\hat{X} \otimes \hat{X} \otimes \hat{X}). \quad (6.9b)$$

Once the block affected by a phase-flip error is identified, applying the operator $\hat{Z} \otimes \hat{Z} \otimes \hat{Z}$ on the block recovers the original state.

Here is a widely known quantum circuit model for Shor's 9-qubit quantum error correction code.

```
In[5]:= qc = QuantumCircuit[
  LogicalForm[Ket[], S@Range[2, 9]],
  CNOT[S[1], S[4]], CNOT[S[1], S[7]],
  S[{1, 4, 7}, 6],
  {CNOT[S[1], S[2]], CNOT[S[4], S[5]], CNOT[S[7], S[8]]},
  {CNOT[S[1], S[3]], CNOT[S[4], S[6]], CNOT[S[7], S[9]]},
  "Invisible" → S@{3.5, 6.5}
]
```



This is the logical basis state $|0\rangle$ of the 9-qubit code.

```
In[6]:= out0 = ExpressionFor@QuantumCircuit[Ket[S[1] → 0], qc];
QuissoFactor@LogicalForm[out0, S@Range[9]]
```

$$\frac{1}{2\sqrt{2}} \left(|\theta_{S_1}\theta_{S_2}\rangle |\theta_{S_3}\rangle + |\theta_{S_1}\theta_{S_2}\rangle |\theta_{S_3}\rangle \right) \otimes \left(|\theta_{S_4}\theta_{S_5}\rangle |\theta_{S_6}\rangle + |\theta_{S_4}\theta_{S_5}\rangle |\theta_{S_6}\rangle \right) \otimes \left(|\theta_{S_7}\theta_{S_8}\rangle |\theta_{S_9}\rangle + |\theta_{S_7}\theta_{S_8}\rangle |\theta_{S_9}\rangle \right)$$

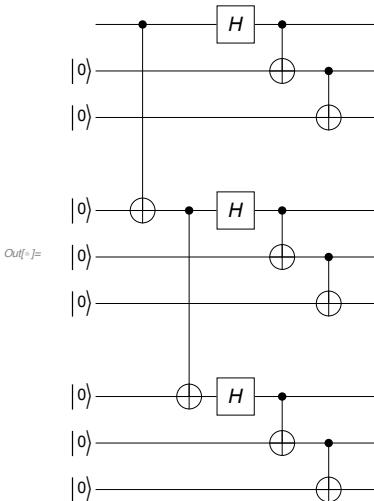
This is the logical basis state $|1\rangle$ of the 9-qubit code.

```
In[7]:= out1 = ExpressionFor@QuantumCircuit[Ket[S[1] → 1], qc];
QuissoFactor@LogicalForm[out1, S@Range[9]];

Out[7]=  $\frac{1}{2\sqrt{2}} (\left|0_{S_1}0_{S_2}\right\rangle \left|0_{S_3}\right\rangle - \left|1_{S_1}1_{S_2}\right\rangle \left|1_{S_3}\right\rangle) \otimes (\left(\left|0_{S_4}0_{S_5}\right\rangle \left|0_{S_6}\right\rangle - \left|1_{S_4}1_{S_5}\right\rangle \left|1_{S_6}\right\rangle\right) \otimes (\left(\left|0_{S_7}0_{S_8}\right\rangle \left|0_{S_9}\right\rangle - \left|1_{S_7}1_{S_8}\right\rangle \left|1_{S_9}\right\rangle))$ 
```

Here is another equivalent implementation of Shor's 9-qubit quantum error correction code.

```
In[8]:= qc = QuantumCircuit[
  LogicalForm[Ket[], S@Range[2, 9]],
  CNOT[S[1], S[4]], CNOT[S[4], S[7]],
  S[{1, 4, 7}, 6],
  {CNOT[S[1], S[2]], CNOT[S[4], S[5]], CNOT[S[7], S[8]]},
  {CNOT[S[2], S[3]], CNOT[S[5], S[6]], CNOT[S[8], S[9]]},
  "Invisible" → S@{3.5, 6.5}
]
```



This is the logical basis state $|0\rangle$ of the 9-qubit code.

```
In[9]:= out0 = ExpressionFor@QuantumCircuit[Ket[S[1] → 0], qc];
QuissoFactor@LogicalForm[out0, S@Range[9]];

Out[9]=  $\frac{1}{2\sqrt{2}} (\left|0_{S_1}0_{S_2}\right\rangle \left|0_{S_3}\right\rangle + \left|1_{S_1}1_{S_2}\right\rangle \left|1_{S_3}\right\rangle) \otimes (\left(\left|0_{S_4}0_{S_5}\right\rangle \left|0_{S_6}\right\rangle + \left|1_{S_4}1_{S_5}\right\rangle \left|1_{S_6}\right\rangle\right) \otimes (\left(\left|0_{S_7}0_{S_8}\right\rangle \left|0_{S_9}\right\rangle + \left|1_{S_7}1_{S_8}\right\rangle \left|1_{S_9}\right\rangle))$ 
```

This is the logical basis state $|1\rangle$ of the 9-qubit code.

```
In[10]:= out1 = ExpressionFor@QuantumCircuit[Ket[S[1] → 1], qc];
QuissoFactor@LogicalForm[out1, S@Range[9]];

Out[10]=  $\frac{1}{2\sqrt{2}} (\left|0_{S_1}0_{S_2}\right\rangle \left|0_{S_3}\right\rangle - \left|1_{S_1}1_{S_2}\right\rangle \left|1_{S_3}\right\rangle) \otimes (\left(\left|0_{S_4}0_{S_5}\right\rangle \left|0_{S_6}\right\rangle - \left|1_{S_4}1_{S_5}\right\rangle \left|1_{S_6}\right\rangle\right) \otimes (\left(\left|0_{S_7}0_{S_8}\right\rangle \left|0_{S_9}\right\rangle - \left|1_{S_7}1_{S_8}\right\rangle \left|1_{S_9}\right\rangle))$ 
```

So far, we have seen that the bit-flip and phase-flip errors on single qubits can be successfully detected and corrected as expected by construction of the code. Now the crucial point is that the same code can correct completely arbitrary errors as long as the error occurs in single qubits. Let $|\psi\rangle = |\bar{0}\rangle c_0 + |\bar{1}\rangle c_1$ be the initial state of the encoded qubits. Suppose that an error occurs in one of the qubits. The error can be efficiently described by a quantum operation \mathcal{F} . Suppose that \mathcal{F} is specified by the Kraus elements \hat{F}_μ . Then, the state corrupted by the error is given by

$$\mathcal{F}(|\psi\rangle\langle\psi|) = \sum_\mu \hat{F}_\mu |\psi\rangle\langle\psi| \hat{F}_\mu^\dagger. \quad (6.10)$$

It is a statistical mixture of different pure states. Let us focus on one of them, say, $\hat{F}_\mu |\psi\rangle$. As the Kraus element \hat{F}_μ is an operator on a single qubit j , it can be expanded in the Pauli operators,

$$\hat{F}_\mu = \hat{I}_j C_{0\mu} + \hat{X}_j C_{1\mu} + \hat{Y}_j C_{2\mu} + \hat{Z}_j C_{3\mu}, \quad (6.11)$$

where $\hat{C}_{\lambda\mu}$ are complex coefficients, and the subscript j attached to the Pauli operators indicate that the operators act on the particular qubit j . Thus, the corruption state $\hat{F}_\mu |\psi\rangle$ is a superposition of the initial state $|\psi\rangle$ without error, the state $\hat{X}_j |\psi\rangle$ affected by the bit-flip error, the state $\hat{Z}_j |\psi\rangle$ affected by the phase-flip error, and the state $\hat{Y}_j |\psi\rangle = i\hat{X}\hat{Z}|\psi\rangle$ affected by both. When the error syndrome is diagnosed, the state collapses to the corresponding state, and the proper recovery operation discussed above brings back the initial state. The same procedure applies to all other Kraus elements of \mathcal{F} .

The example clearly demonstrates that once the code can correct just bit-flip and phase-flip errors, then it can correct any arbitrary error. In other words, even though the errors in qubits form a continuum, the fate of a quantum error-correction code is determined by its ability to correct a discrete set of errors. This is established more rigorously in the next section.

6.2 Quantum Error Correction

The characteristic property of quantum information that distinguishes it from classical information is that it can take an arbitrary *continuous* superposition of logical states. Naturally, errors in quantum information form a continuum as any slightest deviation in the superposition results in an error. At a first glance, it may cast immense difficulties even in detecting errors. Ironically, “continuous” quantum information bears a striking difference from information stored in classical analog systems, and errors in quantum information can be “discretized” so to speak. In this section, we will show this in two stages. We first discuss the conditions for errors to be corrected. We then proceed to show that correcting merely a discrete set of errors suffices to correct the continuum of errors in quantum information.

6.2.1 Quantum Error-Correction Conditions

We first present the *quantum error-correction conditions*. These conditions allow to determine, through simple tests, whether a quantum code protects against a certain class of quantum noises. Consider a quantum code associated with the code space \mathcal{V} . We describe the error with a quantum operation \mathcal{E} specified by Kraus elements $\{\hat{E}_\mu\}$ —see Section 5.1.1. We want to show^{6.1} that there exists an error-correction or recovery operation \mathcal{R} that corrects \mathcal{E} on \mathcal{V} if and only if every pair of the Kraus elements satisfy

$$\hat{P}\hat{E}_\mu^\dagger\hat{E}_\nu\hat{P} = A_{\mu\nu}\hat{P} \quad (6.12)$$

for some Hermitian matrix $A_{\mu\nu}$, where \hat{P} is the projector onto \mathcal{V} . In the course, physical meaning of the condition (6.12) will become clear.

Suppose that the condition in (6.12) is satisfied. We can construct the recovery operation \mathcal{R} . As the matrix A is Hermitian by assumption, it can be diagonalized into the form

$$A = U\Lambda U^\dagger \quad (6.13)$$

for some unitary matrix U and diagonal matrix Λ . Define new Kraus elements $\hat{F}_\nu = \sum_\mu \hat{E}_\mu U_{\mu\nu}$, then the condition (6.12) reads as

$$\hat{P}\hat{F}_\mu^\dagger\hat{F}_\nu\hat{P} = \delta_{\mu\nu}\lambda_\nu\hat{P}, \quad (6.14)$$

where λ_μ are the diagonal elements of Λ . Physically, this means that different Kraus elements \hat{F}_μ brings the code space \mathcal{V} to mutually orthogonal subspaces, $\hat{F}_\mu\mathcal{V} \perp \hat{F}_\nu\mathcal{V}$ for $\mu \neq \nu$. Therefore, measuring the error syndrome corresponds to identifying one of the subspaces, and the subsequent recovery procedure corresponds to bringing the subspace back to the code space. Mathematically, each subspace can be identified by finding the projection operator onto it. To find the projection operators, we note that polar decomposition of $\hat{F}_\mu\hat{P}$ leads to

$$\hat{F}_\mu\hat{P} = \hat{V}_\mu\sqrt{\hat{P}\hat{F}_\mu^\dagger\hat{F}_\mu\hat{P}} = \hat{V}_\mu\sqrt{\lambda_\mu\hat{P}} = \sqrt{\lambda_\mu}\hat{V}_\mu\hat{P}, \quad (6.15)$$

where \hat{V}_μ is a unitary operator. We define

$$\hat{P}_\mu := \hat{V}_\mu\hat{P}\hat{V}_\mu^\dagger, \quad (6.16)$$

and assert that they are indeed the desired projection operators

$$\hat{P}_\mu\hat{P}_\nu = \hat{P}_\mu^\dagger\hat{P}_\nu = \frac{\hat{V}_\mu\hat{P}\hat{F}_\mu^\dagger\hat{F}_\nu\hat{P}\hat{V}_\mu^\dagger}{\sqrt{\lambda_\mu\lambda_\nu}} = \delta_{\mu\nu}\hat{P}_\nu. \quad (6.17)$$

Now error syndrome can be diagnosed by the projection measurement (Postulate 3') associated with the projectors \hat{P}_μ . After the diagnosis of the syndrome, one can

^{6.1}We follow the proof in Nielsen & Chuang (2011, Section 10.3).

simply apply the unitary operator \hat{V}_μ^\dagger on the resulting state to recover the initial state.

We have constructed the recovery procedure starting from the condition in (6.12) assumed on the error process. To show that the condition is most general, suppose that the error described by \mathcal{E} is correctable. It means that there exists a quantum operation \mathcal{R} that describes the recovery process. Mathematically, it can be specified by the Kraus elements $\{\hat{R}_\nu\}$ such that

$$\mathcal{R}(\mathcal{E}(\hat{P}\hat{\rho}\hat{P})) = c\hat{P}\hat{\rho}\hat{P} \quad (6.18)$$

for some constant $0 < c \leq 1$. In terms of the Kraus elements of \mathcal{E} and \mathcal{R} , it reads as

$$\sum_{\mu\nu} \hat{R}_\mu \hat{E}_\nu \hat{P} \hat{\rho} (\hat{R}_\mu \hat{E}_\nu \hat{P})^\dagger = c\hat{P}\hat{\rho}\hat{P} \quad (6.19)$$

for all $\hat{\rho}$. The set $\{\hat{R}_\mu \hat{E}_\nu \hat{P}\}$ of Kraus elements define the composite quantum operation $\mathcal{R} \circ \mathcal{E}$ describing recovery after error. Equation (6.19) indicates that the same quantum operation $\mathcal{R} \circ \mathcal{E}$ is described by a single Kraus element $\sqrt{c}\hat{P}$. According to the unitary freedom for Kraus elements discussed in Section 5.1.1 especially Eqs. (5.24), there must exist complex numbers $W_{\mu\nu}$ such that

$$\hat{R}_\mu \hat{E}_\nu = \sqrt{c}\hat{P}W_{\mu\nu} \quad (6.20)$$

and

$$\sum_{\mu\nu} W_{\mu\nu}^* W_{\mu\nu} = 1. \quad (6.21)$$

Therefore, it follows that

$$\sum_\lambda (\hat{R}_\lambda \hat{E}_\mu \hat{P})^\dagger (\hat{R}_\lambda \hat{E}_\nu \hat{P}) = \sum_\lambda \hat{P} \hat{E}_\mu^\dagger \hat{R}_\lambda^\dagger \hat{R}_\lambda \hat{E}_\nu \hat{P} = \hat{P} \hat{E}_\mu^\dagger \hat{E}_\nu \hat{P} = c(W^\dagger W)_{\mu\nu} \hat{P}, \quad (6.22)$$

where we have used the fact $\sum_\lambda \hat{R}_\lambda^\dagger \hat{R}_\lambda = \hat{I}$ because \mathcal{R} is trace-preserving. Since the matrix $A := cW^\dagger W$ is certainly Hermitian by construction, the last equality in (6.22) asserts that the Kraus elements $\{\hat{E}_\mu\}$ satisfy the condition (6.12).

6.2.2 Discretization of errors

Once a quantum code is proven to protect against a particular type of quantum noise in accordance with the quantum error-correction conditions, it is robust against a surprisingly wide class of quantum noises. In fact, it turns out that if a quantum noise operation \mathcal{F} with the error operators $\{\hat{E}_\mu\}$ can be corrected by a quantum code \mathcal{V} , then the code also protects against any quantum noise \mathcal{F}' with the error operators $\{\hat{F}_\nu\}$ that are linear superpositions of $\{\hat{E}_\mu\}$,

$$\hat{F}_\nu = \sum_\mu \hat{E}_\mu M_{\mu\nu}, \quad (6.23)$$

where $M_{\mu\nu}$ is a matrix of complex numbers.

The proof is simple. As the quantum noise \mathcal{E} is correctable on the code \mathcal{V} , it must satisfy the quantum error-correction conditions in (6.12). Multiply the equations with the Matrix M^\dagger and M ,

$$\sum_{\mu'\nu'} \hat{P} M_{\mu\mu'}^\dagger \hat{E}_{\mu'}^\dagger \hat{E}_{\nu'} M_{\nu'\nu} \hat{P} = \sum_{\mu'\nu'} M_{\mu\mu'}^\dagger A_{\mu'\nu'} M_{\nu'\nu} \hat{P} \quad (6.24)$$

It leads to

$$\hat{P} \hat{F}_\mu^\dagger \hat{F}_\nu \hat{P} = (M^\dagger A M)_{\mu\nu} \hat{P}, \quad (6.25)$$

which is formally the same as the quantum error-correction conditions. Therefore, the quantum noise \mathcal{F} must be correctable on \mathcal{V} as well.

Quite simple to prove, the implication of the statement is huge: Consider a class of errors on single qubits. Suppose that it is described by a quantum operation \mathcal{E} specified by Kraus elements $\{\hat{E}_{j\mu}\}$, where the index j indicates the qubit subject to the error and μ denotes different error processes. As $\hat{E}_{j\mu}$ are operators on the single qubit j , they can be expanded in terms of the Pauli operators \hat{S}_j^0 , \hat{S}_j^x , \hat{S}_j^y , and \hat{S}_j^z . In order to check if a given quantum error-correction code protects against arbitrary single-qubit errors, one has only to inspect the condition

$$\hat{P} \hat{S}_j^\mu \hat{S}_j^\nu \hat{P} = A_{\mu\nu} \hat{P} \quad (6.26)$$

for all j and a Hermitian matrix A . In other words, when one constructs a quantum error-correction code, it is sufficient to check a finite (and hence discrete) set of conditions to make it sure that the code protects against arbitrary single-qubit errors.

6.3 Stabilizer Formalism

There are several ways to construct a quantum error-correction code. One of the most popular ways is to take the analogy of the classical linear codes. The Calderbank-Shor-Steane (CSS) codes are an example. In this book, we skip the approach and will discuss stabilizer codes. Stabilizer codes do not allow a direct counterpart in classical codes, not to speak of its elegance and simplicity. Before we discuss stabilizer codes in the next section, we introduce the basic framework of stabilizer formalism. Stabilizer formalism was first put forward by [Gottesman \(1996\)](#). The formalism is explained in great detail in [Gottesman \(1997\)](#) and presented also in [Nielsen & Chuang \(2011\)](#). Recently, stabilizer formalism has been extended to the measurement-based quantum computation scheme ([Browne & Briegel, 2016](#)).

A quantum error-correction code encodes quantum states of a fixed number of qubits in a subspace of the larger Hilbert space of more qubits. The subspace is called the *code space* of the particular code. The code space is specified by the

logical basis states that span it. We have seen some examples, Eqs. (6.1), (6.4), and (6.7) in Section 6.1. A natural way to analyze the code is to inspect how the quantum states in the code space are affected by errors. Mathematically, it is achieved by examining the evolution of quantum states under the quantum operation describing the error process as in the previous section, Section 6.2. This corresponds to the usual Schrödinger picture of dynamical processes in quantum mechanics.

It may sound odd at first glance, but a code space can also be specified by operators that preserve it, and errors can be described by tracking those operators. This task is what stabilizer formalism is for. In a broad sense, it is analogous to the Heisenberg picture of dynamical processes in quantum mechanics.

Let \mathcal{G} be a group of operators acting on quantum states of a quantum system—see Appendix C for a brief introduction to group theory. In quantum computation, the most relevant group is the unitary group $U(2^n)$ of quantum logic gate operators on n qubits. Let $|\psi\rangle$ be a quantum state. When it remains unchanged under an operator \hat{G} in the group \mathcal{G} ,

$$\hat{G} |\psi\rangle = |\psi\rangle, \quad (6.27)$$

the state is said to be *stabilized* by \hat{G} , or equivalently, \hat{G} is said to *stabilize* $|\psi\rangle$. For example, the Bell state $|\Phi\rangle := (|00\rangle + |11\rangle)/\sqrt{2}$ is stabilized by $\hat{X} \otimes \hat{X}$. The set of operators that stabilize a quantum state $|\psi\rangle$ forms a subgroup of \mathcal{G} . The subgroup is called the *stabilizer subgroup* or simply *stabilizer* of the quantum state $|\psi\rangle$. For example, the Bell state $|\Phi\rangle$ is also stabilized by $\hat{Z} \otimes \hat{Z}$ and $-\hat{Y} \otimes \hat{Y}$ as well as, trivially, by $\hat{I} \otimes \hat{I}$. These operators form a subgroup of $U(2^2)$. In other words, the subgroup

$$\{\hat{I} \otimes \hat{I}, \hat{X} \otimes \hat{X}, \hat{Z} \otimes \hat{Z}, -\hat{Y} \otimes \hat{Y}\} \quad (6.28)$$

is the stabilizer of the Bell state $|\Phi\rangle$.

Consider one of the Bell states.

```
In[1]:= ket = BellState[S@{1, 2}, 3];
ket // LogicalForm
Out[1]= 
$$\frac{|0_{S_1}0_{S_2}\rangle - |1_{S_1}1_{S_2}\rangle}{\sqrt{2}}$$

```



```
In[2]:= grp = {1, -S[1, 1] ** S[2, 1], S[1, 3] ** S[2, 3], S[1, 2] ** S[2, 2]};
PauliForm[grp]
Out[2]= {I ⊗ I, -(X ⊗ X), Z ⊗ Z, Y ⊗ Y}
```



```
In[3]:= new = grp ** ket;
new // LogicalForm
Out[3]= 
$$\left\{ \frac{|0_{S_1}0_{S_2}\rangle}{\sqrt{2}} - \frac{|1_{S_1}1_{S_2}\rangle}{\sqrt{2}}, \frac{|0_{S_1}0_{S_2}\rangle}{\sqrt{2}} - \frac{|1_{S_1}1_{S_2}\rangle}{\sqrt{2}}, \frac{|0_{S_1}0_{S_2}\rangle}{\sqrt{2}} - \frac{|1_{S_1}1_{S_2}\rangle}{\sqrt{2}}, \frac{|0_{S_1}0_{S_2}\rangle}{\sqrt{2}} - \frac{|1_{S_1}1_{S_2}\rangle}{\sqrt{2}} \right\}$$

```

The notion of stabilizer is extended to subspaces. Let \mathcal{V} be a subspace of the Hilbert space associated with a quantum system. The stabilizer \mathcal{S} of \mathcal{V} is the set of all operators \hat{G} in \mathcal{G} that stabilize all quantum states in \mathcal{V} . For example, consider the code space \mathcal{V} of the bit-flip correction code discussed in Section 6.1.1. \mathcal{V} is spanned by the two logical basis states $|000\rangle$ and $|111\rangle$. The stabilizer \mathcal{S} of \mathcal{V} is given by

$$\mathcal{S} = \{\hat{I} \otimes \hat{I} \otimes \hat{I}, \hat{Z} \otimes \hat{Z} \otimes \hat{I}, \hat{I} \otimes \hat{Z} \otimes \hat{Z}, \hat{Z} \otimes \hat{I} \otimes \hat{Z}\}. \quad (6.29)$$

Consider the code space \mathcal{V} of the bit-flip correction code. It is spanned by the following two logical basis states.

```
In[7]:= ss = S[{1, 2, 3}, None];
bs = {Ket[], Ket[S@{1, 2, 3} \rightarrow 1]};
bs // LogicalForm
Out[7]= { |0s1 0s2 0s3\rangle, |1s1 1s2 1s3\rangle }
```

The stabilizer \mathcal{S} of \mathcal{V} is given as follows.

```
In[8]:= grp = {1, S[1, 3] ** S[2, 3], S[2, 3] ** S[3, 3], S[1, 3] ** S[3, 3]};
grp // PauliForm
Out[8]= { I \otimes I \otimes I, Z \otimes Z \otimes I, I \otimes Z \otimes Z, Z \otimes I \otimes Z }
```

This checks the above group indeed stabilizes \mathcal{V} .

```
In[9]:= grp ** bs[[1]] // LogicalForm[#, ss] &
grp ** bs[[2]] // LogicalForm[#, ss] &
Out[9]= { |0s1 0s2 0s3\rangle, |0s1 0s2 0s3\rangle, |0s1 0s2 0s3\rangle, |0s1 0s2 0s3\rangle }
Out[10]= { |1s1 1s2 1s3\rangle, |1s1 1s2 1s3\rangle, |1s1 1s2 1s3\rangle, |1s1 1s2 1s3\rangle }
```

Consider the code space of the phase-flip correction code. It is spanned by the following two logical basis states.

```
In[11]:= ss = S[{1, 2, 3}, None];
bs =
{ProductState[ss \rightarrow {1, 1} / Sqrt[2]], ProductState[ss \rightarrow {1, -1} / Sqrt[2]]];
bs // LogicalForm
Out[11]= { \left( \frac{|0\rangle}{\sqrt{2}} + \frac{|1\rangle}{\sqrt{2}} \right)_{s_1} \otimes \left( \frac{|0\rangle}{\sqrt{2}} + \frac{|1\rangle}{\sqrt{2}} \right)_{s_2} \otimes \left( \frac{|0\rangle}{\sqrt{2}} + \frac{|1\rangle}{\sqrt{2}} \right)_{s_3}, \left( \frac{|0\rangle}{\sqrt{2}} - \frac{|1\rangle}{\sqrt{2}} \right)_{s_1} \otimes \left( \frac{|0\rangle}{\sqrt{2}} - \frac{|1\rangle}{\sqrt{2}} \right)_{s_2} \otimes \left( \frac{|0\rangle}{\sqrt{2}} - \frac{|1\rangle}{\sqrt{2}} \right)_{s_3} }
```

The stabilizer \mathcal{S} of \mathcal{V} is given as follows.

```
In[12]:= grp = {1, S[1, 1] ** S[2, 1], S[2, 1] ** S[3, 1], S[1, 1] ** S[3, 1]};
grp // PauliForm
Out[12]= { I \otimes I \otimes I, X \otimes X \otimes I, I \otimes X \otimes X, X \otimes I \otimes X }
```

This checks the above group indeed stabilizes \mathcal{V} .

```
In[=]:= grp ** bs[[1]] // Elaborate // QuissoFactor // LogicalForm[#, ss] &
Out[=]:= { (|0S10S20S3> + |0S10S21S3>) (|0S10S20S3> + |0S11S20S3>) (|0S10S20S3> + |1S10S20S3>),
           2 √2 ,
( |0S10S20S3> + |0S10S21S3>) (|0S10S20S3> + |0S11S20S3>) (|0S10S20S3> + |1S10S20S3>),
           2 √2 ,
( |0S10S20S3> + |0S10S21S3>) (|0S10S20S3> + |0S11S20S3>) (|0S10S20S3> + |1S10S20S3>),
           2 √2 ,
( |0S10S20S3> + |0S10S21S3>) (|0S10S20S3> + |0S11S20S3>) (|0S10S20S3> + |1S10S20S3>)
           2 √2 }

In[=]:= grp ** bs[[2]] // Elaborate // QuissoFactor // LogicalForm[#, ss] &
Out[=]:= { (|0S10S20S3> - |0S10S21S3>) (|0S10S20S3> - |0S11S20S3>) (|0S10S20S3> - |1S10S20S3>),
           2 √2 ,
( |0S10S20S3> - |0S10S21S3>) (|0S10S20S3> - |0S11S20S3>) (|0S10S20S3> - |1S10S20S3>),
           2 √2 ,
( |0S10S20S3> - |0S10S21S3>) (|0S10S20S3> - |0S11S20S3>) (|0S10S20S3> - |1S10S20S3>),
           2 √2 ,
( |0S10S20S3> - |0S10S21S3>) (|0S10S20S3> - |0S11S20S3>) (|0S10S20S3> - |1S10S20S3>)
           2 √2 }
```

Under a quantum operation, a given subspace evolves to a different subspace, and hence the stabilizer of the subspace changes accordingly. In stabilizer formalism, one describes the quantum operation by tracking the stabilizer rather than the subspace or quantum states contained in it. For a general parent group \mathcal{G} such as the unitary group $U(2^n)$, the stabilizer formalism may involve unnecessary mathematical complications. However, in many quantum information processing procedures including quantum error correction, the Pauli operators and their tensor products are sufficient. In this case, stabilizer formalism turns out to be extremely simple and insightful.

6.3.1 Pauli Group

In general, finding the stabilizer of a given subspace and vice versa may be non-trivial. Fortunately, when operators are restricted to the Pauli operators or their tensor products on a fixed number n of qubits, stabilizers have particularly simple structures and useful properties. Formally, such operators form a group (Appendix C), called the *Pauli group* $\mathcal{P}(n)$ on n qubits. Moreover, in accordance with the discretization of errors discussed in Section 6.2, such operators are enough to describe any quantum noise operation. Therefore, stabilizer subgroups of the Pauli group provide convenient and powerful mathematical tools to construct and implement a wide class of quantum error-correction codes. In the remaining of the book, we assume that the parent group of any stabilizer is a Pauli group.

The Pauli group on a single qubit consists of the single-qubit Pauli operators. Elisting the elements, it is given by

$$\mathcal{P}(1) := \{\pm \hat{I}, \pm i\hat{I}, \pm \hat{X}, \pm i\hat{X}, \pm \hat{Y}, \pm i\hat{Y}, \pm \hat{Z}, \pm i\hat{Z}\}. \quad (6.30)$$

The additional phase factors ± 1 and $\pm i$ are required because the product of two Pauli operators gives another Pauli operator accompanied by factor $\pm i$. Otherwise, the underlying set is not closed under the group multiplication. The Pauli group on n qubits is given by

$$\mathcal{P}(n) := \{\pm 1, \pm i\} \times \{\hat{I}, \hat{X}, \hat{Y}, \hat{Z}\}^{\otimes n}. \quad (6.31)$$

We call an element of the Pauli group on n qubits an n -qubit *Pauli operators*.

As the number of elements in the Pauli group grows fast with qubits, 4^{n+1} , it is convenient to describe the group in terms of *generators* (Definition C.2). For example, the single-qubit Pauli group is generated by \hat{X} , \hat{Y} , and \hat{Z} (or by $i\hat{I}$, \hat{X} , and \hat{Z}). We denote the generators by enclosing them in $\langle \{\cdots\} \rangle$. Whence one can write

$$\mathcal{P}(1) = \langle \{\hat{X}, \hat{Y}, \hat{Z}\} \rangle. \quad (6.32)$$

The Pauli group on n qubits are generated by tensor products of the Pauli operators acting non-trivially on one and only one qubit. For example, we observe that

$$\mathcal{P}(2) = \langle \{\hat{X} \otimes \hat{I}, \hat{Y} \otimes \hat{I}, \hat{Z} \otimes \hat{I}, \hat{I} \otimes \hat{X}, \hat{I} \otimes \hat{Y}, \hat{I} \otimes \hat{Z}\} \rangle \quad (6.33)$$

for the Pauli group on two qubits. We will often omit the number n of qubits from the Pauli group specification $\mathcal{P}(n)$ and write just \mathcal{P} when there is no risk of confusion.

Here is the Pauli group on a single qubit.

```
In[1]:= grp = PauliGroup[1]
Out[1]= PauliGroup[1]

In[2]:= elm = GroupElements[grp];
PauliForm[elm]
Out[2]= {I, X, Y, Z, -I, -X, -Y, -Z, i I, i X, i Y, i Z, -i I, -i X, -i Y, -i Z}
```

The number of elements quickly increases with the number of qubits.

```
In[3]:= grp = PauliGroup[2]
Out[3]= PauliGroup[2]

In[4]:= elm = GroupElements[grp];
PauliForm[elm]
Out[4]= {I ⊗ I, I ⊗ X, I ⊗ Y, I ⊗ Z, X ⊗ I, X ⊗ X, X ⊗ Y, X ⊗ Z, Y ⊗ I, Y ⊗ X, Y ⊗ Y, Y ⊗ Z,
Z ⊗ I, Z ⊗ X, Z ⊗ Y, Z ⊗ Z, -(I ⊗ I), -(I ⊗ X), -(I ⊗ Y), -(I ⊗ Z), -(X ⊗ I),
-(X ⊗ X), -(X ⊗ Y), -(X ⊗ Z), -(Y ⊗ I), -(Y ⊗ X), -(Y ⊗ Y), -(Y ⊗ Z), -(Z ⊗ I),
-(Z ⊗ X), -(Z ⊗ Y), -(Z ⊗ Z), i I ⊗ I, i I ⊗ X, i I ⊗ Y, i I ⊗ Z, i X ⊗ I, i X ⊗ X,
i X ⊗ Y, i X ⊗ Z, i Y ⊗ I, i Y ⊗ X, i Y ⊗ Y, i Y ⊗ Z, i Z ⊗ I, i Z ⊗ X, i Z ⊗ Y, i Z ⊗ Z,
-i I ⊗ I, -i I ⊗ X, -i I ⊗ Y, -i I ⊗ Z, -i X ⊗ I, -i X ⊗ X, -i X ⊗ Y, -i X ⊗ Z,
-i Y ⊗ I, -i Y ⊗ X, -i Y ⊗ Y, -i Y ⊗ Z, -i Z ⊗ I, -i Z ⊗ X, -i Z ⊗ Y, -i Z ⊗ Z}
```

```
In[5]:= GroupOrder@grp
Out[5]= 64
```

Groups can be conveniently described by the use of generators.

```
In[6]:= grp = PauliGroup[1]
Out[6]= PauliGroup[1]
```

For example, the single-qubit Pauli group is generated by three Pauli operators.

```
In[7]:= gnr = GroupGenerators[grp];
PauliForm[gnr]
Out[7]= {X, Y, Z}
```

Here is a generating set of the Pauli group on two qubits. Six generators are enough to generate it.

```
In[8]:= gnr = GroupGenerators[PauliGroup[2]];
PauliForm[gnr]
Out[8]= {I ⊗ X, I ⊗ Y, I ⊗ Z, X ⊗ I, Y ⊗ I, Z ⊗ I}
```

Operators in Pauli groups have two extremely useful properties that eventually determine the group-theoretical structure of stabilizers and stabilizer codes. First, any two elements of a Pauli group either commute or anti-commute with each other. Second, every element of the Pauli group squares to $\pm \hat{I}$. The elements that square to \hat{I} have eigenvalues ± 1 whereas those that square to $-\hat{I}$ have eigenvalues $\pm i$.

Pauli group has two extremely useful properties. These are illustrated here.

To avoid the irrelevant phase factors ± 1 and $\pm i$, we play with simple tensor products of single-qubit Pauli operators rather than the full Pauli group.

```
In[9]:= ops = Flatten@Outer[Multiply, S[1, Full], S[2, Full]]
Out[9]= {S20, S2X, S2Y, S2Z, S1X, S1X S2X, S1Y S2Y, S1Z S2Z, S1Y, S1Y S2X, S1Y S2Y, S1Y S2Z, S1Z, S1Z S2X, S1Z S2Y, S1Z S2Z}
In[10]:= PauliForm[ops]
Out[10]= {I ⊗ I, I ⊗ X, I ⊗ Y, I ⊗ Z, X ⊗ I, X ⊗ X, X ⊗ Y,
X ⊗ Z, Y ⊗ I, Y ⊗ X, Y ⊗ Y, Y ⊗ Z, Z ⊗ I, Z ⊗ X, Z ⊗ Y, Z ⊗ Z}
```

This table shows that the elements of the Pauli group either commute or anti-commute with each other.

```
In[11]:= mat = Outer[GottesmanTest, ops, ops];
TableForm[mat[[-6 ;;, -6 ;]],
TableHeadings → {PauliForm@ops[[-6 ;;]], PauliForm@ops[[-6 ;;]]},
TableAlignments → Right]
```

	Y ⊗ Y	Y ⊗ Z	Z ⊗ I	Z ⊗ X	Z ⊗ Y	Z ⊗ Z
Y ⊗ Y	1	-1	-1	1	-1	1
Y ⊗ Z	-1	1	-1	1	1	-1
Z ⊗ I	-1	-1	1	1	1	1
Z ⊗ X	1	1	1	1	-1	-1
Z ⊗ Y	-1	1	1	-1	1	-1
Z ⊗ Z	1	-1	1	-1	-1	1

The rule is also simple that allows to determine which elements commute and which elements anti-commute .

Any element of the Pauli group squares to ± 1 . Elements with the phase factor $\pm i$ squares to -1 .

```
In[7]:= elm = GroupElements[PauliGroup[S@{1, 2}]];
sqr = Map[#**# &, elm];
TableForm[{PauliForm[elm] [[ -8 ; ;]], sqr [[ -8 ; ;]]}]
Out[7]/TableForm=

$$\begin{array}{cccccccc}
-i Y \otimes I & -i Y \otimes X & -i Y \otimes Y & -i Y \otimes Z & -i Z \otimes I & -i Z \otimes X & -i Z \otimes Y & -i Z \otimes Z \\
-1 & -1 & -1 & -1 & -1 & -1 & -1 & -1
\end{array}$$

```

The above property implies that any element of the Pauli group has eigenvalues ± 1 or $\pm i$.

In most applications of Pauli group, the explicit multiplication table is not necessary. Sufficient is commutation (or anti-commutation) of elements. As long as commutation of elements is concerned, one can ignore the phase factors ± 1 and $\pm i$ in the elements of Pauli group. Mathematically, this can be achieved by considering the *factor group*

$$\mathcal{P}'(n) := \mathcal{P}(n)/\mathcal{Z}(n), \quad (6.34)$$

where

$$\mathcal{Z}(n) := \{\pm \hat{I}^{\otimes n}, \pm i \hat{I}^{\otimes n}\} \quad (6.35)$$

is an invariant subgroup of $\mathcal{P}(n)$.^{6.2} For example, in the factor group

$$\mathcal{P}' = \{\{\pm \hat{I}, \pm i \hat{I}\}, \{\pm \hat{X}, \pm i \hat{X}\}, \{\pm \hat{Y}, \pm i \hat{Y}\}, \{\pm \hat{Z}, \pm i \hat{Z}\}\} \quad (6.36)$$

on a single qubit, the elements $\pm \hat{X}$ and $\pm i \hat{X}$ are not distinguished any longer. They are regarded equivalent and denoted collectively by the *coset*

$$\hat{X}\mathcal{Z} = \{\pm \hat{X}, \pm i \hat{X}\} \quad (6.37)$$

or, equivalently, $\mathcal{Z}\hat{X}$. This leads to a drastically simple structure of \mathcal{P}' compared with \mathcal{P} . For example, $\hat{X}\hat{Y}$ is now equivalent to $\hat{Y}\hat{X}$, that is,

$$\hat{X}\hat{Z}\mathcal{Z} = \hat{Z}\hat{X}\mathcal{Z}. \quad (6.38)$$

It means that unlike the genuine Pauli group $\mathcal{P}(n)$, the factor group $\mathcal{P}'(n)$ is Abelian and isomorphic to $(\mathbb{Z}_2)^{2n}$.

For later use, let us establish an isomorphism (i.e., one-to-one correspondence) explicitly between $\mathcal{P}'(n)$ and $(\mathbb{Z}_2)^{2n}$. It suffices that on each qubit, we associate each Pauli operator with an ordered pair in $\mathbb{Z}_2 \times \mathbb{Z}_2$ as follows

$$\hat{I} \leftrightarrow (0, 0), \quad \hat{X} \leftrightarrow (1, 0), \quad \hat{Z} \leftrightarrow (0, 1), \quad \hat{Y} \leftrightarrow (1, 1). \quad (6.39)$$

^{6.2}In fact, $\mathcal{Z}(n)$ is the center of $\mathcal{P}(n)$, the largest Abelian invariant subgroup of $\mathcal{P}(n)$ —see Appendix C.

For example, given a tensor product $\hat{G} = i\hat{X} \otimes \hat{Z} \otimes \hat{I} \otimes \hat{Y}$, on a system of four qubits, the coset $\hat{G}\mathcal{Z}$ corresponds to $(1, 0; 0, 1; 0, 0; 1, 1)$, that is,

$$\hat{G}\mathcal{Z} \leftrightarrow (1, 0; 0, 1; 0, 0; 1, 1). \quad (6.40)$$

Recall that the Abelian group $(\mathbb{Z}_2)^{2n}$ with bit-wise addition modulo 2 as the group multiplication can also be regarded as a vector space over the field \mathbb{Z}_2 of binary numbers—see Appendix C.5. For this reason, it is convenient to denote the correspondence in (6.40) by

$$|\hat{G}\rangle := (1, 0; 0, 1; 0, 0; 1, 1)^T, \quad \langle \hat{G}| := (1, 0; 0, 1; 0, 0; 1, 1). \quad (6.41)$$

We call the vector $|\hat{G}\rangle$ in $(\mathbb{Z}_2)^{2n}$ the *Gottesman vector* of the operator \hat{G} (more precisely, of the coset $\hat{G}\mathcal{Z}$).

Ignoring the phase factors in the operators of Pauli group, we are dealing with the factor group of the Pauli group with respect to the center $\{\mathbb{I}, -\mathbb{I}, i\mathbb{I}, -i\mathbb{I}\}$.

```
In[1]:= op = -I * S[1, 2] ** S[3, 1] ** S[4, 3]
PauliForm[op, ss = S@{1, 2, 3, 4}]

Out[1]= - I S1^y S3^x S4^z

Out[1]= - I Y ⊗ I ⊗ X ⊗ Z
```

This gives the Gottesman vector of the above operator (more precisely, the coset represented by the operator).

```
In[2]:= vec = GottesmanVector[op, ss]
Out[2]= {1, 1, 0, 0, 1, 0, 0, 1}
```

An operator representing the coset is recovered.

```
In[3]:= new = FromGottesmanVector[vec, ss]
Out[3]= S1^y S3^x S4^z
```

The correspondence between Pauli operators (more precisely, cosets in \mathcal{P}') and Gottesman vectors is a group isomorphism—it preserves the group multiplication. Specifically, the multiplication of Pauli operators corresponds to the addition (modulo 2) of the corresponding Gottesman vectors. This provides a convenient way to check independence of Pauli operators: Suppose that the elements $\hat{G}_1, \hat{G}_2, \dots, \hat{G}_k$ of $\mathcal{P}(n)$ are *independent*—none of them can be obtained (up to a phase factor) by a multiplication of the others. Then, it follows—see Problem 6.2—that the corresponding Gottesman vectors $|\hat{G}_1\rangle, |\hat{G}_2\rangle, \dots, |\hat{G}_k\rangle$ are *linearly independent* of each other in $(\mathbb{Z}_2)^{2n}$. The converse is also true—if the Gottesman vectors $|\hat{G}_1\rangle, |\hat{G}_2\rangle, \dots, |\hat{G}_k\rangle$ are linearly independent of each other in $(\mathbb{Z}_2)^{2n}$, then the operators $\hat{G}_1, \hat{G}_2, \dots, \hat{G}_k$ are independent (up to a phase factor) of each other in $\mathcal{P}(n)$. This method is especially useful when one wants to pick up independent generators from an over-generating set of a stabilizer.

The correspondence also carries the commutation relation of Pauli operators to a certain type of relation of the corresponding Gottesman vectors. Let \hat{G}_1 and

\hat{G}_2 be n -qubit Pauli operators. It is straightforward to show by inspection that if they commute with each other, then

$$\langle \hat{G}_1 | \hat{J} | \hat{G}_2 \rangle = 0, \quad (6.42)$$

where the operator \hat{J} on Gottesman vectors is defined by

$$\hat{J} := \bigoplus_{j=1}^n \begin{bmatrix} 0 & 1 \\ 1 & 0 \end{bmatrix} = \begin{bmatrix} 0 & 1 & & & \\ 1 & 0 & & & \\ & & \ddots & & \\ & & & 0 & 1 \\ & & & 1 & 0 \end{bmatrix}. \quad (6.43)$$

If \hat{G}_1 and \hat{G}_2 anti-commute with each other, then

$$\langle \hat{G}_1 | \hat{J} | \hat{G}_2 \rangle = 1. \quad (6.44)$$

The converses of the above statements also hold. The operator \hat{J} equips the space $(\mathbb{Z}_2)^{2n}$ of Gottesman vectors with an inner product.

One can use Gottesman vectors to prove another interesting property of Pauli groups: Consider again a set of independent Pauli operators $\hat{G}_1, \hat{G}_2, \dots, \hat{G}_k$. Then, there exists an operator \hat{G} in $\mathcal{P}(n)$ such that \hat{G} anti-commutes with one and only one of $\hat{G}_1, \hat{G}_2, \dots, \hat{G}_k$ but commutes with the rest—Problem 6.3. For example, consider the two independent operators $\hat{G}_1 := \hat{Z} \otimes \hat{Z} \otimes \hat{I}$ and $\hat{G}_2 := \hat{I} \otimes \hat{Z} \otimes \hat{Z}$. The operator $\hat{G} := \hat{X} \otimes \hat{I} \otimes \hat{I}$ anti-commutes with \hat{G}_1 but commutes with \hat{G}_2 . On the other hand, the operator $\hat{G}' := \hat{I} \otimes \hat{Z} \otimes \hat{Y}$ commutes with \hat{G}_1 and anti-commutes with \hat{G}_2 .

6.3.2 Properties of Stabilizers

The simple structure and useful properties of Pauli groups enable us to inspect many aspects of stabilizers without peculiar specifications. Here we discuss and summarize general properties of stabilizers to be exploited repeatedly later.

An immediate consequence of the properties of Pauli groups in Section 6.3.1—in particular that any two elements of a Pauli group either commutes or anti-commutes—is that all elements of a stabilizer must commute with each other (Problem 6.1). That is, any stabilizer is an *Abelian* subgroup of the Pauli group. Recall also that every elements of a Pauli group squares to $\pm \hat{I}$. If an operator in a Pauli group squares to $-\hat{I}$, then its eigenvalue must be $\pm i$ and it can stabilize no state. Putting these observations together, we can set down the necessary conditions for a subgroup \mathcal{S} of the Pauli group to stabilize a non-trivial subspace \mathcal{V} in a compact form: (i) all elements of \mathcal{S} commute, and (ii) \mathcal{S} does not include $-\hat{I}$ as an element.

Let us now inspect the relation between a stabilizer and the subspace stabilized by it more closely: Suppose that \mathcal{S} is a stabilizer on n qubits generated by k

independent operators $\hat{G}_1, \dots, \hat{G}_k$. Since all elements in the stabilizer commute each other and square to \hat{I} , \mathcal{S} is essentially the same as $(\mathbb{Z}_2)^k$. Mathematically, \mathcal{S} is isomorphic to the Abelian group $(\mathbb{Z}_2)^k$ with the group multiplication given by the addition modulo 2; see Appendix C.5.

The states stabilized by \mathcal{S} are simultaneous eigenvectors of the generators of \mathcal{S} . The more generators \mathcal{S} has, the more constraints the states have to satisfy. Naturally, the bigger the number of generators of \mathcal{S} , the smaller the dimension of the subspace \mathcal{V} stabilized by \mathcal{S} is. More explicitly, the proposition is that the dimension of \mathcal{V} is equal to 2^{n-k} . In other words, the subspace \mathcal{V} can encode $(n-k)$ logical qubits. It is one of the most basic properties that enable stabilizer codes as we will see in Section 6.4 below. To see it, note that any n -qubit Pauli operator, say \hat{G}_1 , without a phase factor $\pm i$ divides the Hilbert space \mathcal{H} into two orthogonal subspaces respectively belonging to the eigenvalues ± 1 . Consequently, the subspace \mathcal{W}_1 belonging to the eigenvalue 1 is the subspace stabilized by \hat{G}_1 and has dimension 2^{n-1} . An additional Pauli operator \hat{G}_2 that commutes \hat{G}_1 divides \mathcal{W}_1 again into two orthogonal subspaces. One of them belonging to the eigenvalue 1, \mathcal{W}_2 , is stabilized by both \hat{G}_1 and \hat{G}_2 . Obviously, \mathcal{W}_2 is 2^{n-2} -dimensional. One can repeat the argument to confirm the proposition. For example, the operator $\hat{Z} \otimes \hat{Z}$ stabilizes any combination of $|00\rangle$ and $|11\rangle$ or, equivalently, any combination of

$$\frac{|00\rangle + |11\rangle}{\sqrt{2}}, \quad \frac{|00\rangle - |11\rangle}{\sqrt{2}}. \quad (6.45)$$

The operator $\hat{X} \otimes \hat{X}$ stabilizes any combination of

$$\frac{|00\rangle + |11\rangle}{\sqrt{2}}, \quad \frac{|01\rangle + |10\rangle}{\sqrt{2}}. \quad (6.46)$$

The vector space stabilized by each of the operators is thus two dimensional. On the other hand, it is only the states proportional to $(|00\rangle + |11\rangle)/\sqrt{2}$ that are stabilized simultaneously by the both operators. Therefore, the subspace stabilized by the stabilizer $\langle\{\hat{X} \otimes \hat{X}, \hat{Z} \otimes \hat{Z}\}\rangle$ is one dimensional. We observe that the operator $\hat{X} \otimes \hat{X}$ divides the subspace spanned by the vectors in (6.45) into two subspaces, one spanned by $(|00\rangle + |11\rangle)/\sqrt{2}$ and the other by $(|00\rangle - |11\rangle)/\sqrt{2}$. The operator $\hat{Z} \otimes \hat{Z}$ divides the subspace spanned by the vector in (6.46) in a similar manner.

One can prove the above proposition more rigorously. Note that the operators $(\hat{I} \pm \hat{G}_j)/2$ for each j are the projectors onto the eigensubspaces of \hat{G}_j belonging to the eigenvalues ± 1 , respectively. We generalize them and define

$$\hat{P}_x := \prod_{j=1}^k \frac{\hat{I} + (-1)^{x_j} \hat{G}_j}{2} \quad (6.47)$$

for each binary string $x := (x_1 x_2 \cdots x_k)_2$. \hat{P}_x is the projector onto the simultaneous eigensubspace of the generators $\hat{G}_1, \dots, \hat{G}_k$ belonging to the eigenvalue

$(-1)^{x_1+\dots+x_k}$. As such, they are orthogonal, that is,

$$\hat{P}_x \hat{P}_y = \delta_{xy} \hat{P}_x \quad (6.48)$$

for all x, y in $(\mathbb{Z}_2)^k$. In particular, \hat{P}_0 is the projector onto the subspace \mathcal{V} stabilized by \mathcal{S} . Previously (Section 6.3.2), we noted that given a set of independent Pauli operators, there exists a operator that anti-commutes with only one operator and commutes with the other. Accordingly, for each x , one can find an operator \hat{G}_x such that

$$\hat{G}_x \hat{P}_{(0,\dots,0)} \hat{G}_x = \hat{P}_x. \quad (6.49)$$

It means that the subspaces $\hat{P}_x \mathcal{H}$ for all x have the same dimension as \mathcal{V} . Finally, note that

$$\sum_{x=0}^{2^k-1} \hat{P}_x = \hat{I} \quad (6.50)$$

because

$$\sum_{x_1, x_2, \dots} (-1)^{x_1+x_2+\dots} = \sum_{x_1} (-1)^{x_1} \sum_{x_2} (-1)^{x_2} \dots = 0. \quad (6.51)$$

Therefore, the 2^k projectors \hat{P}_x divide the 2^n -dimensional Hilbert space \mathcal{H} into orthogonal subspaces of the same dimension. The dimension of each subspace must be $2^n/2^k = 2^{n-k}$.

We have seen above that a subspace \mathcal{V} can be identified by its stabilizer \mathcal{S} . For quantum information processing within the subspace \mathcal{V} , one also needs to specify a logical basis of \mathcal{V} . In stabilizer formalism, logical basis states are specified indirectly through *logical operators*. Let us first take an example. Consider again the subspace \mathcal{V} stabilized by $\mathcal{S} = \langle \{\hat{Z} \otimes \hat{Z}\} \rangle$. Suppose that we choose the two states

$$|\bar{0}\rangle = \frac{|00\rangle + |11\rangle}{\sqrt{2}}, \quad |\bar{1}\rangle = \frac{|00\rangle - |11\rangle}{\sqrt{2}}, \quad (6.52)$$

as the logical basis states. Within \mathcal{V} , the operator $\hat{X} \otimes \hat{X}$ has eigenstates $|\bar{0}\rangle$ and $|\bar{1}\rangle$ with eigenvalues ± 1 , respectively, and hence behaves like the Pauli Z operator on a single qubit. In this sense, we put $\bar{Z} = \hat{X} \otimes \hat{X}$ and regard it as the *logical* Pauli Z operator. On the other hand, $\hat{Z} \otimes \hat{I}$ flips $|\bar{0}\rangle$ to $|\bar{1}\rangle$ and vice versa. Further, it fixes the relative phase between the basis states. In thesees senses, it behaves like the Pauli X operator on a single qubit. We thus put $\bar{X} = \hat{Z} \otimes \hat{I}$ and regard it as the logical Pauli X operator. Note that \bar{Z} and \bar{X} commute with all elements of the stabilizer \mathcal{S} and hence preserve the stabilized subspace \mathcal{V} . Nevertheless, unlike the operators in the stabilizer \mathcal{S} , they act non-trivially on \mathcal{V} —they move around the quantum states inside \mathcal{V} .

More generally, the logical basis states $|\bar{x}\rangle$ associated with binary strings $\bar{x} := (\bar{x}_1 \bar{x}_2 \dots \bar{x}_{n-k})_2$ can be designated by two sets of logical operators $\bar{Z}_1, \dots, \bar{Z}_{n-k}$ and

$\bar{X}_1, \dots, \bar{X}_{n-k}$. While measurement of the logical Pauli Z operators \bar{Z}_j distinguishes the logical basis states, that is,

$$\bar{Z}_j |\bar{x}\rangle = \bar{x}_j |\bar{x}\rangle, \quad (6.53)$$

the logical Pauli X operators \bar{X}_j fix the relative phases of the logical basis states by imposing

$$\bar{X}_j |\bar{x}_1, \dots, \bar{x}_j, \dots, \bar{x}_{n-k}\rangle = |\bar{x}_1, \dots, \bar{x}_j \oplus 1, \dots, \bar{x}_{n-k}\rangle. \quad (6.54)$$

Of course, given a stabilizer, one can construct logical operators without referring to the logical basis states. The necessary condition is that the logical operators should preserve the given subspace \mathcal{V} , transforming states within \mathcal{V} . It implies that any logical operator, \bar{Z}_j or \bar{X}_j , must commute with all generators of the stabilizer \mathcal{S} . Nevertheless, the logical operators must not belong to the stabilizer—they are independent of the generators of the stabilizer. Otherwise, their operations on \mathcal{V} are all trivial. Note that for a given logical basis, the choice of the logical operators is not unique. For example, for the same basis logical states in (6.52), one can choose $\bar{Z} = -\hat{Y} \otimes \hat{Y}$ and $\bar{X} = \hat{I} \otimes \bar{Z}$. Mathematically, equivalent logical operators belong to the same coset with respect to \mathcal{S} . Given a set of independent generators of the stabilizer, there is a systematic method to construct logical operators utilizing Gottesman vectors. We refer interested readers to Cleve & Gottesman (1997).

6.3.3 Unitary Gates in Stabilizer Formalism

Consider a unitary operation \hat{U} on a quantum system. Suppose that \mathcal{S} stabilizes a given subspace \mathcal{V} . The unitary operation \hat{U} transforms \mathcal{V} onto $\hat{U}\mathcal{V}$, and \mathcal{S} does not stabilize the new subspace any longer. Then, what is the stabilizer of the new subspace $\hat{U}\mathcal{V}$?

Suppose that an operator \hat{G} is an element of the stabilizer \mathcal{S} . Then, $\hat{G}|\psi\rangle = |\psi\rangle$ for any quantum state $|\psi\rangle$ in the subspace \mathcal{V} . We observe that

$$\hat{U}\hat{G}\hat{U}^\dagger \hat{U}|\psi\rangle = \hat{U}\hat{G}|\psi\rangle = \hat{U}|\psi\rangle. \quad (6.55)$$

It means that $\hat{U}\hat{G}\hat{U}^\dagger$ stabilizes $\hat{U}|\psi\rangle$, and hence the whole subspace $\hat{U}\mathcal{V}$. In other words, the group $\hat{U}\mathcal{S}\hat{U}^\dagger$ is the stabilizer of the new subspace $\hat{U}\mathcal{V}$. Now suppose that $\{\hat{G}_j\}$ is a generating set of the stabilizer \mathcal{S} . As any element of \mathcal{S} is generated by \hat{G}_j , any operator of the form $\hat{U}\hat{G}\hat{U}^\dagger$ for $\hat{G} \in \mathcal{S}$ is generated by $\hat{U}\hat{G}_j\hat{U}^\dagger$. That is, the new stabilizer $\hat{U}\mathcal{S}\hat{U}^\dagger$ is generated by $\hat{U}\hat{G}_j\hat{U}^\dagger$. To sum up, under a unitary operation \hat{U} , the generators \hat{G}_j of the stabilizer are transformed as follows

$$\hat{G}_j \mapsto \hat{U}\hat{G}_j\hat{U}^\dagger. \quad (6.56)$$

As the unitary operation \hat{U} changes logical basis states, the logical operators must transform accordingly. Following the above line of arguments for the generators of the stabilizer, we see that the logical operators also transform in the same manner

$$\bar{X}_j \mapsto \hat{U}\bar{X}_j\hat{U}^\dagger, \quad \bar{Z}_j \mapsto \hat{U}\bar{Z}_j\hat{U}^\dagger. \quad (6.57)$$

So far, the discussion was completely general. However, we are mainly interested in stabilizer subgroups of the Pauli group. For an element \hat{G} of the Pauli group, in general, $\hat{U}\hat{G}\hat{U}^\dagger$ does not belong to the Pauli group and cannot be expressed with a tensor product of the single-qubit Pauli operators. It would be useful to focus on unitary operators \hat{U} the conjugation by which transforms tensor products of the single-qubit Pauli operators to such products. This is the subject of the next subsection.

6.3.4 Clifford Group

The *Clifford group* $\mathcal{C}(n)$ on n qubits is the group of unitary operators \hat{U} that leave the Pauli group $\mathcal{P}(n)$ invariant under conjugation, that is,

$$\mathcal{C}(n) := \{\hat{U} : \hat{U}\mathcal{P}(n)\hat{U}^\dagger = \mathcal{P}(n)\}. \quad (6.58)$$

Mathematically, the Clifford group is the *normalizer* of the Pauli group in the unitary group $U(2^n)$. Elements of the Clifford group on n qubits are called n -qubit *Clifford operators*. As for Pauli group, we will often omit the number n of qubits from the Clifford group specification $\mathcal{C}(n)$ and write \mathcal{C} when there is no risk of confusion.

The Clifford group was first introduced in 1998 by Daniel Gottesman ([Gottesman, 1998](#)) when he reformulated the quantum error-correction codes in terms of stabilizers. Clifford group is useful and interesting since stabilizers remain subgroups of the Pauli group under conjugation by its elements.

The Clifford group $\mathcal{C}(n)$ is generated by Hadamard gates and quadrant phase gates^{6.3} on individual qubits, and CNOT (or equivalently CZ) gates on all pairs of qubits. To see this, first consider single-qubit transformations in the Clifford group $\mathcal{C}(1)$. By definition, the *conjugation* by a single-qubit operator in $\mathcal{C}(1)$ permutes the Pauli operators \hat{X} , \hat{Y} and \hat{Z} up to a phase factor—note that the conjugation by a single-qubit unitary operator cannot alter the identity operator \hat{I} . Mathematically, it is equivalent to an one-to-one mapping $\mathbb{Z}_4 \rightarrow \mathbb{Z}_4$ with 0 fixed.^{6.4} Geometrically, it can be regarded as an axes-permuting rotation in the Bloch space—see Fig. 6.2 (b). Single-qubit transformations in the Clifford group thus correspond to the six symmetry transformations in the *point group*

$$D_3 := \{\hat{I}, \hat{M}_x, \hat{M}_y, \hat{M}_z, \hat{R}, \hat{R}^2\}, \quad (6.59)$$

^{6.3}The quadrant phase gate is defined in Eq. (2.28).

^{6.4}Here $\mathbb{Z}_m := \{0, 1, \dots, m - 1\}$ is merely a set of m elements without any further structure. The notation is more commonly used for a *cyclic group* of order m as in Eq. (C.10).

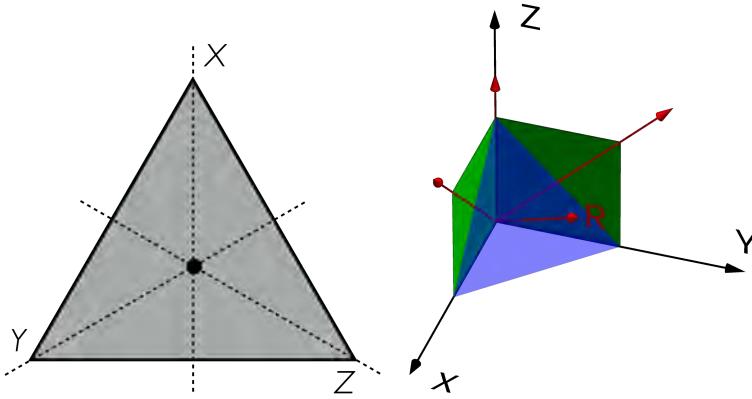


Figure 6.2: (a) An equilateral triangle, which is invariant under the symmetry transformations in the point group D_3 . The dashed lines indicate the mirror planes in which the object is symmetric. The point in the middle indicates the rotation by angle $2\pi/3$. (b) Rotations in the Bloch space that permute the axes.

which leaves an equilateral triangle invariant—see Fig. 6.2 (a). Here \hat{M}_x , \hat{M}_y , and \hat{M}_z denote the mirror reflections about the planes through the vertex X, Y, and Z, respectively, of the triangle. \hat{R} represents the rotation by angle $2\pi/3$ around the axis through the center of the triangle. For example, consider the Hadamard gate. It is a rotation by angle π around the axis $(1, 0, 1)$ in the Bloch space. It follows from Eq. (2.25) (or a simple inspection) that the Hadamard gate exchanges \hat{X} and \hat{Z} , $\hat{H}\hat{X}\hat{H}^\dagger = \hat{Z}$ and $\hat{H}\hat{Z}\hat{H}^\dagger = \hat{X}$, while it keeps \hat{Y} intact, $\hat{H}\hat{Y}\hat{H}^\dagger = -\hat{Y}$. In short,

$$\hat{H} : (\hat{X}, \hat{Y}, \hat{Z}) \mapsto (\hat{Z}, -\hat{Y}, \hat{X}), \quad (6.60)$$

and the conjugation by \hat{H} corresponds to the symmetry transformation \hat{M}_y in D_3 . Similarly, the conjugation by the quadrant phase gate \hat{Q} exchanges \hat{X} and \hat{Y} ,

$$\hat{Q} : (\hat{X}, \hat{Y}, \hat{Z}) \mapsto (\hat{Y}, -\hat{X}, \hat{Z}), \quad (6.61)$$

and is equivalent to the symmetry transformation \hat{M}_z in D_3 . Changing axes before and after \hat{M}_y , it follows from Eq. (2.25) that \hat{M}_x is equivalent to $\hat{Q}\hat{H}\hat{Q}^\dagger$. Finally, the equivalent of \hat{R} in D_3 is the rotation by $2\pi/3$ around the axis $(1, 1, 1)$ in the Bloch space. It is identical to $\hat{H}\hat{Q}^\dagger$. Overall, the correspondence between the single-qubit Clifford group and the point group D_3 is summarized as

$$\hat{M}_z \leftrightarrow \hat{Q}, \quad \hat{M}_y \leftrightarrow \hat{H}, \quad \hat{M}_x \leftrightarrow \hat{Q}\hat{H}\hat{Q}^\dagger, \quad \hat{R} \leftrightarrow \hat{H}\hat{Q}^\dagger. \quad (6.62)$$

This asserts that the single-qubit Clifford group is generated by \hat{H} and \hat{Q} .

The single-qubit Clifford group is rather simple. It is generated by the Hadamard gate and the quadrant phase gate.

```
In[7]:= gnr = GroupGenerators@CliffordGroup[S]
Out[7]= {SH, SS}
```

In fact, it is equivalent to the point group D_3 . The point group consists of the six symmetry transformation $\{\mathbf{I}, \mathbf{M}_x, \mathbf{M}_y, \mathbf{M}_z, \mathbf{R}, \mathbf{R} \star \mathbf{R}\}$.

```
My = Elaborate@S[6];
Mz = Elaborate@S[7];
Mx = Mz ** My ** Dagger[Mz];
R = My ** Dagger[Mz];
```

Note that the *conjugation* (not to be confused with conjugate) by an operator is a special case of supermap with a single unitary Kraus element.

```
Conjugation[op_] := Supermap[op]
```

Take a look how the Pauli operators transform under the conjugation by single-qubit operations in the Clifford group.

```
In[8]:= ops = {S[1], S[2], S[3]};
Thread[ops → Conjugation[Mx]@ops] // PauliForm // TableForm
Thread[ops → Conjugation[My]@ops] // PauliForm // TableForm
Thread[ops → Conjugation[Mz]@ops] // PauliForm // TableForm

Out[8]/TableForm=
X → -X
Y → Z
Z → Y

Out[8]/TableForm=
X → Z
Y → -Y
Z → X

Out[8]/TableForm=
X → Y
Y → -X
Z → Z

In[9]:= Thread[ops → Conjugation[R]@ops] // PauliForm // TableForm
Thread[ops → Conjugation[R ** R]@ops] // PauliForm // TableForm

Out[9]/TableForm=
X → Y
Y → Z
Z → X

Out[9]/TableForm=
X → Z
Y → X
Z → Y
```

Let us move on, and examine two-qubit operations in the Clifford group $\mathcal{C}(2)$. The conjugation by a two-qubit operation transforms a pair of four Pauli operators $\hat{I}, \hat{X}, \hat{Y}, \hat{Z}$ to another pair, just like a one-to-one mapping $\mathbb{Z}_4 \times \mathbb{Z}_4 \rightarrow \mathbb{Z}_4 \times \mathbb{Z}_4$. Apparently, local combinations of Hadamard and quadrant phase gates cannot transform a pair including \hat{I} to a pair that does not, nor vice versa. For example, the mapping

$$\hat{Y} \otimes \hat{I} \mapsto \hat{Z} \otimes \hat{Y} \quad (6.63)$$

is impossible with local combinations of Hadamard and quadrant phase gates alone since

$$(\hat{U} \otimes \hat{V})(\hat{Y} \otimes \hat{I})(\hat{U} \otimes \hat{V})^\dagger = (\hat{U}\hat{Y}\hat{U}^\dagger) \otimes \hat{I} \quad (6.64)$$

for all unitary operators \hat{U} and \hat{V} . This is where CNOT gate plays a critical role. The conjugation by the CNOT gate controlled by the first qubit on the second qubit maps

$$\begin{aligned}\hat{X} \otimes \hat{I} &\mapsto \hat{X} \otimes \hat{X}, \\ \hat{Z} \otimes \hat{I} &\mapsto \hat{Z} \otimes \hat{I}, \\ \hat{I} \otimes \hat{X} &\mapsto \hat{I} \otimes \hat{X}, \\ \hat{I} \otimes \hat{Z} &\mapsto \hat{Z} \otimes \hat{Z}.\end{aligned}\tag{6.65}$$

It is interesting to note that under the CNOT gate, the Pauli X operator “propagates” *forward* (from the control to target qubit) while the Pauli Z operator propagates *backward* (from the target to control qubit). The mappings of pairs involving \hat{Y} can be deduced from the above correspondence using the identity $\hat{Y} = i\hat{X}\hat{Z}$. Since a product of Pauli operators can be mapped to any product of them as long as the product do not involve \hat{I} , a single CNOT gate is sufficient to complement the actions of Hadamard and quadrant phase gates. For example, one way to achieve the transformation in Eq. (6.63) is applying successively the conjugations by $\hat{Q}^\dagger \otimes \hat{I}$, CNOT, $\hat{H} \otimes \hat{Q}$ leading to

$$\hat{Y} \otimes \hat{I} \rightarrow \hat{X} \otimes \hat{I} \rightarrow \hat{X} \otimes \hat{X} \rightarrow \hat{Z} \otimes \hat{Y}.\tag{6.66}$$

The same arguments apply for any number of qubits, and Hadamard, quadrant phase, and CNOT gates are sufficient to generate any operation in the Clifford group.

Here demonstrated is the correspondence between one-to-one mappings $\mathbb{Z}_4 \times \mathbb{Z}_4 \rightarrow \mathbb{Z}_4 \times \mathbb{Z}_4$ and the two-qubit operations in the Clifford group.

```
In[1]:= gnr = CNOT[S[1], S[2]]  
Out[1]= CNOT[{S1}, {S2}]
```

Under the conjugation by the CNOT gate, the Pauli operators transform as the following.

```
In[2]:= ops = Multiply @@@ Tuples@{S[1, Full], S[2, Full]};  
Timing[new = Elaborate@Thread[ops → Conjugation[gnr]@ops];]  
new[[2 ;; ; ; 4]] // PauliForm // TableForm  
Out[2]= {0.187539, Null}  
Out[3]= TableForm[  
 {I ⊗ X → I ⊗ X,  
 X ⊗ X → X ⊗ I,  
 Y ⊗ X → Y ⊗ I,  
 Z ⊗ X → Z ⊗ X}]
```

Combinations of the CNOT, Hadamard, and quadrant phase gates generate all transformations preserving the Pauli group. Here is one example.

```
gnr = CNOT[S[1], S[2]] ** S[1, 6] ** Dagger[S[1, 7]] ** S[2, 7] // Elaborate;  
ops = {S[1, 1], S[2, 1], S[1, 3], S[2, 3]};
```

```
In[=]:= Thread[ops → Conjugation[gnr]@ops] // PauliForm // TableForm
Out[=]:= {{X ⊗ I → Y ⊗ X}, {I ⊗ X → Z ⊗ Y}, {Z ⊗ I → X ⊗ Y}, {I ⊗ Z → Z ⊗ Z}}
```

Let us now prove rigorously that a Clifford group is generated by Hadamard and quadrant phase gates on single qubits and CNOT gates on pairs of qubits: The inductive proof is constructive, and turns out to be useful to find implementations of operations in the Clifford group (Gottesman, 1998).

Let \hat{U} be an element of the $(n+1)$ -qubit Clifford group $\mathcal{C}(n+1)$. Without loss of generality, one can assume that

$$\hat{U}(\hat{Z} \otimes \hat{J})\hat{U}^\dagger = \hat{X} \otimes \hat{A}, \quad \hat{U}(\hat{X} \otimes \hat{J})\hat{U}^\dagger = \hat{Z} \otimes \hat{B}, \quad (6.67)$$

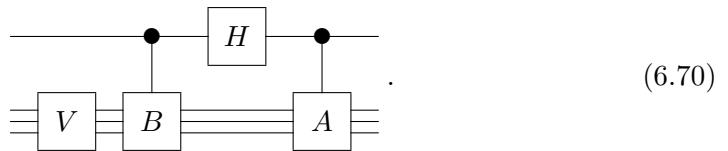
where $\hat{J} := \hat{I}^{\otimes n}$, and \hat{A} and \hat{B} are products of the Pauli operators, $\hat{A}, \hat{B} \in \mathcal{P}(n)$. If \hat{U} does not satisfy the above properties, one can always rearrange qubits and apply additional single-qubit operations so that the modified operation assume the forms—see Problem 6.6. It is also possible to assume that

$$\hat{U}(\hat{Z} \otimes \hat{J})\hat{U}^\dagger = \hat{X} \otimes \hat{A}, \quad \hat{U}(\hat{X} \otimes \hat{J})\hat{U}^\dagger = \hat{I} \otimes \hat{B}, \quad (6.68)$$

but here we will exploit the forms in (6.67). Define an operator \hat{V} acting on n qubits by

$$\hat{V}|y\rangle := \sum_{x=0}^{2^n-1} |x\rangle (\langle 0| \otimes \langle x|) \hat{U}(|0\rangle \otimes |y\rangle) \quad (6.69)$$

for $y = 0, 1, 2, \dots, 2^n - 1$. In effect, \hat{V} projects out of \hat{U} the component corresponding to $|1\rangle$ on the first qubit. One can show that derived from \hat{U} , the new operator \hat{V} is an element of the n -qubit Clifford group $\mathcal{C}(n)$ —see Problem 6.7. In other words, the conjugation by \hat{V} preserves the n -qubit Pauli group $\mathcal{P}(n)$. The goal is then to express \hat{U} in terms of \hat{A} , \hat{B} , and \hat{V} . In short, \hat{U} can be implemented in the following quantum circuit model



As \hat{A} and \hat{B} are products of the Pauli operators, the controlled- \hat{A} and controlled- \hat{B} gate can be implemented with Hadamard, quadrant phase, and CNOT gates. Therefore, \hat{U} can be achieved in a similar manner as long as \hat{V} is.

The only remainig is to show that \hat{U} is equivalent to the quantum circuit model in (6.70). Let us consider a general $(n+1)$ -qubit state

$$|\Psi\rangle = \frac{|0\rangle \otimes |\alpha\rangle + |1\rangle \otimes |\beta\rangle}{\sqrt{2}}, \quad (6.71)$$

where $|\alpha\rangle$ and $|\beta\rangle$ are n -qubit states. Upon the operation of \hat{U} on it, the first term transforms to

$$\hat{U}(|0\rangle \otimes |\alpha\rangle) = |0\rangle \otimes |\alpha_0\rangle + |1\rangle \otimes |\alpha_1\rangle. \quad (6.72)$$

As $\hat{Z} \otimes \hat{J}$ does not alter the first term, we also obtain

$$\begin{aligned} \hat{U}(|0\rangle \otimes \hat{\alpha}) &= \hat{U}(\hat{Z} \otimes \hat{J})\hat{U}^\dagger\hat{U}(|0\rangle \otimes |\alpha\rangle) \\ &= (\hat{X} \otimes \hat{A})(|0\rangle \otimes |\alpha_0\rangle + |1\rangle \otimes |\alpha_1\rangle) \\ &= |0\rangle \otimes \hat{A}|\alpha_1\rangle + |1\rangle \otimes \hat{A}|\alpha_0\rangle. \end{aligned} \quad (6.73)$$

As the two results in (6.72) and (6.73) must be identical, the two states $|\alpha_0\rangle$ and $|\alpha_1\rangle$ should be related to each other by \hat{A}

$$\hat{A}|\alpha_0\rangle = |\alpha_1\rangle, \quad \hat{A}|\alpha_1\rangle = |\alpha_0\rangle. \quad (6.74)$$

In passing, these relations are consistent with the fact that \hat{A} is a product of the Pauli operators. Putting (6.74) back into (6.72),

$$\hat{U}(|0\rangle \otimes |\alpha\rangle) = |0\rangle \otimes |\alpha_0\rangle + |1\rangle \otimes \hat{A}|\alpha_1\rangle = (\hat{I} \otimes \hat{J} + \hat{X} \otimes \hat{A})(|0\rangle \otimes \hat{V}|\alpha\rangle), \quad (6.75)$$

where for the second equality, we have used the fact $|\alpha_0\rangle = \hat{V}|\alpha\rangle$ by definition of the operator \hat{V} in (6.69).

Next, we examine how the second term transforms under the action of \hat{U} . We observe that

$$\begin{aligned} \hat{U}(|1\rangle \otimes |\beta\rangle) &= \hat{U}(\hat{X} \otimes \hat{J})(|0\rangle \otimes |\beta\rangle) = (\hat{Z} \otimes \hat{B})\hat{U}(|0\rangle \otimes |\beta\rangle) \\ &= (\hat{Z} \otimes \hat{B})(\hat{I} \otimes \hat{J} + \hat{X} \otimes \hat{A})(|0\rangle \otimes \hat{V}|\beta\rangle), \end{aligned} \quad (6.76)$$

where we have used (6.75) for the last equality. We note that $\hat{X} \otimes \hat{A}$ and $\hat{Z} \otimes \hat{B}$ anti-commute since $\hat{U}(\hat{Z} \otimes \hat{J})\hat{U}^\dagger$ and $\hat{U}(\hat{X} \otimes \hat{J})\hat{U}^\dagger$ obviously do. This leads to

$$\begin{aligned} \hat{U}(|1\rangle \otimes |\beta\rangle) &= (\hat{I} \otimes \hat{J} - \hat{X} \otimes \hat{A})(\hat{Z} \otimes \hat{B})(|0\rangle \otimes \hat{V}|\beta\rangle) \\ &= (\hat{I} \otimes \hat{J} - \hat{X} \otimes \hat{A})(|0\rangle \otimes \hat{B}\hat{V}|\beta\rangle). \end{aligned} \quad (6.77)$$

Putting (6.75) and (6.77) together, we have

$$\hat{U}|\Psi\rangle = \frac{|0\rangle \otimes \hat{V}|\alpha\rangle + |1\rangle \otimes \hat{A}\hat{V}|\alpha\rangle + |0\rangle \otimes \hat{B}\hat{V}|\beta\rangle + |1\rangle \otimes \hat{A}\hat{B}\hat{V}|\beta\rangle}{\sqrt{2}}, \quad (6.78)$$

which is exactly the result produced by the quantum circuit model in (6.70)

The peculiarity of the Clifford group is most pronounced in the *Gottesman-Knill theorem* (Gottesman, 1999). It states that so-called *Clifford circuits*,

^{6.5}The Gottesman-Knill theorem is more general and also applies to so-called *stabilizer circuits*. In addition to Clifford gates, a stabilizer circuit includes Clifford gates conditioned on the outcomes of the measurement of Pauli operators.

quantum circuits consisting of unitary operators from the Clifford group, can be efficiently simulated on a classical computer. It sounds surprising because Clifford operators are sufficient to generate a rich range of quantum effects including the Greenberger-Horne-Zeilinger (GHZ) experiment (Greenberger *et al.*, 1989), quantum teleportation, and super dense coding. They are also sufficient to encode and decode quantum error-correction codes (Section 6.4). Nevertheless, the theorem indicates that Clifford group “falls short of the full power of quantum computation.” Indeed, we recall that in addition to Clifford gates, we need octant phase gates or Toffoli gates to implement an arbitrary unitary gate to an arbitrary accuracy (Section 2.4).

How is it possible to simulate Clifford circuits efficiently on a classical computer? The key point is that stabilizer formalism tracks the generators of the stabilizer rather than the quantum states as in the usual methods. To follow the evolution of quantum states, one needs to keep track of an exponential number of complex amplitudes. However, for a system of n qubits, there are an order of n generators of the stabilizer because any finite group \mathcal{G} has a generating set of at most $\log_2 |\mathcal{G}|$ elements (Theorem C.2). Each generator takes $2n$ bits for the Pauli X or Y operators on n qubits and an additional bit for the phase factor ± 1 . Overall, one needs a order of n^2 bits to track the evolution of the stabilizer. We have seen that the Clifford group is generated by the Hadamard, octant phase, and CNOT gates and that the generators under the conjugate by these elementary gates can be updated in polynomial time, $\mathcal{O}(n)$ time for gate.

A specific algorithm to simulate Clifford circuits was proposed in Aaronson & Gottesman (2004). Recently, also proposed was a simulation algorithm for fast simulation of stabilizer circuits using graph states.

6.3.5 Measurements in Stabilizer Formalism

Upon the measurement of an observable \hat{M} , a quantum state $|\psi\rangle$ “collapses” to one of the eigenstates of \hat{M} . When \hat{M} is a tensor product of the Pauli operators, $\hat{M} \in \mathcal{P}(n)$, it has two eigenvalues ± 1 . The measurement process is described by the projectors $\hat{P}_{\pm} = (\hat{I} \pm \hat{M})/2$ onto the eigenspaces belonging to the eigenvalues ± 1 . The description so far has been in the Schrödinger picture. How do the stabilizer and the logical operators change under the measurement in the stabilizer formalism?

Suppose that \mathcal{S} is the stabilizer of a subspace \mathcal{V} of our interest. Recall from Section 6.3.1 that since the observable \hat{M} is a member of the Pauli group $\mathcal{P}(n)$, it either commutes or anti-commutes with any specific operator in the stabilizer \mathcal{S} . It leaves two possibilities, \hat{M} either anti-commutes with some elements of \mathcal{S} or commutes with every element of \mathcal{S} . Let us first examine the former case: We designate by \hat{G} one of the elements that anti-commute with \hat{M} . The post-measurement state becomes $\hat{P}_{\pm} |\psi\rangle$ depending on the result ± 1 of the measurement.

We note that

$$\hat{G}\hat{P}_-|\psi\rangle = \frac{1}{2}\hat{G}(\hat{I} - \hat{M})|\psi\rangle = \frac{1}{2}(\hat{I} + \hat{M})\hat{G}|\psi\rangle = \hat{P}_+|\psi\rangle. \quad (6.79)$$

In other words, after the measurement of \hat{M} , we can make it sure that the post-measurement state is always $\hat{P}_+|\psi\rangle$, if necessary, by applying \hat{G} . With this practice assumed, the relevant subspace after the measurement is now $\hat{P}_+\mathcal{V}$. For any state $|\psi\rangle$ in \mathcal{V} , the post-measurement state $\hat{P}_+|\psi\rangle$ is an eigenstate of \hat{M} belonging to the eigenvalue +1. It means that \hat{M} is an element of the stabilizer \mathcal{S}' of the new subspace $\hat{P}_+\mathcal{V}$. On the other hand, the operator \hat{G} cannot belong to \mathcal{S}' any longer. Any element of \mathcal{S} is still a member of \mathcal{S}' as long as it commutes with \hat{M} . If some element \hat{G}' in \mathcal{S} does not commute with \hat{M} , then $\hat{G}\hat{G}'$ does and hence is a member of \mathcal{S}' . It is physically important to note that by construction, the observable \hat{M} commutes with all elements of \mathcal{S}' .

Now let us turn to the case where \hat{M} commutes with every member of \mathcal{S} . We have to distinguish further two sub-cases. First, \hat{M} itself belongs to \mathcal{S} . In this case, the measurement of \hat{M} gives the outcome +1 with unit probability, and nothing happens to the quantum states nor the stabilizer. The remaining case is when \hat{M} commutes with all elements of \mathcal{S} and yet does not belong to \mathcal{S} . In this case, measuring \hat{M} reveals some information of the quantum state, and the subspace \mathcal{V} cannot maintain its structure—the stabilizer formalism breaks down in some sense. For example, suppose that we measure one of the logical operators, say, $\hat{M} = \bar{Z}$. If the state is $|\psi\rangle = |\bar{0}\rangle c_0 + |\bar{1}\rangle c_1$, then measuring \bar{Z} causes $|\psi\rangle$ to collapse to either $|\bar{0}\rangle$ or $|\bar{1}\rangle$, breaking the encoding. Such a measurement must be performed only at the end of whole computation. Such an observable \hat{M} may occur as a Kraus element of error process, rather than a pure measurement. Then it corresponds to an unrecoverable error.

The new logical operators can be obtained in a similar way. If they commute with \hat{M} , they remain unaffected. If there is any logical operator \bar{L} that does not commute with \hat{M} , then it can be replaced with $\hat{G}\bar{L}$ as the latter certainly commutes with \hat{M} . It is straightforward to check that the new logical operators constructed this way satisfy the same commutation relations as the original logical operators do, and they commute with all elements in the new stabilizer \mathcal{S}' .

To summarize, one can track the evolution of the stabilizer and the logical operators of a given subspace under the measurement of an observable \hat{M} according to the following rules:

1. First, identify a generator \hat{G} of the stabilizer \mathcal{S} that anti-commutes with \hat{M} .
2. Next, pick up new generators of the stabilizer. First, replace \hat{G} with \hat{M} in the list of generators of \mathcal{S} . Second, as for each \hat{G}' of other generators of \mathcal{S} , keep it if it commutes with \hat{M} , and replace it with $\hat{G}\hat{G}'$ if it anti-commutes with \hat{M} .

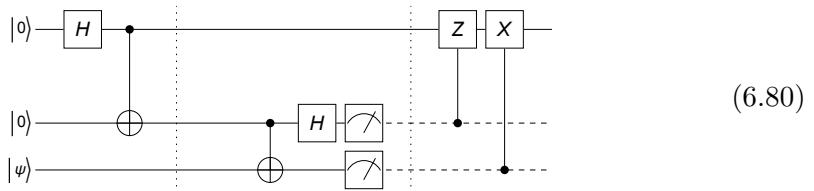
3. Finally, we choose a new set of logical operators. For each logical operator \bar{L} (\bar{X}_j or \bar{Z}_j), keep it if it commutes with \hat{M} , but replace it with $\hat{G}\bar{L}$ if it anti-commutes with \hat{M} .

Recall that the rules assume that after the measurement of the observable \hat{M} , the generator \hat{G} is operated on the system conditioned on the measurement outcome. This way, the post-measurement state remains to be stabilized by the newly constructed stabilizer regardless of the measurement outcome. As the operation of \hat{G} is conditioned on classical data (the measurement outcome), it is analogous to classical feedback control. In this sense we call it a *semiclassical* feedback control.

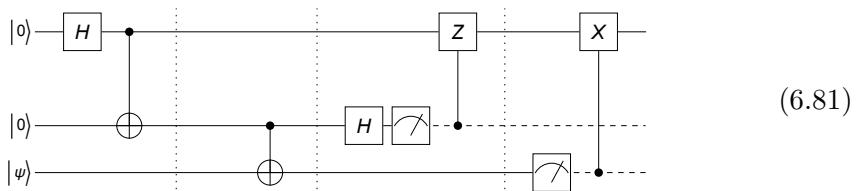
6.3.6 Examples

The above analyses of unitary gates and measurements in stabilizer formalism are useful to analyze what happens upon a combination of specific measurements and operations. In particular, if a part of the quantum register starts in a known quantum state, for example, initialized in $|0\rangle \otimes |0\rangle \otimes \cdots \otimes |0\rangle$, it can often be described by a stabilizer. Let us take two heuristic examples, which were worked out in detail in [Gottesman \(1999\)](#).

The first example is quantum teleportation discussed in Section 4.1 and summarized in the following quantum circuit model



The protocol is implemented in three stages which are distinguished by the vertical dashed lines in the quantum circuit model. The first stage is to share the entangled pair of qubits. The second in the middle corresponds to the Bell measurement. In the final stage, the measurement outcomes are transmitted to Alice, and Alice adjusts the state of her qubit according to the classical information. In order to reinterpret the quantum teleportation protocol in stabilizer formalism, we rearrange the quantum circuit model in (6.80) into



At the point marked by the first vertical line, the relevant stabilizer \mathcal{S} is generated by

$$\hat{G}_1 = \hat{X} \otimes \hat{X} \otimes \hat{I}, \quad \hat{G}_2 = \hat{Z} \otimes \hat{Z} \otimes \hat{I}. \quad (6.82)$$

We pick up two logical operators

$$\bar{X} = \hat{I} \otimes \hat{I} \otimes \hat{X}, \quad \bar{Z} = \hat{I} \otimes \hat{I} \otimes \hat{Z}. \quad (6.83)$$

One can convince oneself that they commute with all the generators (and hence all elements) of the stabilizer. Under the CNOT gate from qubit B to C , the generators and the logical operators are transformed as

$$\begin{cases} \hat{G}_1 \mapsto \hat{X} \otimes \hat{X} \otimes \hat{X}, \\ \hat{G}_2 \mapsto \hat{Z} \otimes \hat{Z} \otimes \hat{I}; \end{cases} \quad \begin{cases} \bar{X} \mapsto \hat{I} \otimes \hat{I} \otimes \hat{X}, \\ \bar{Z} \mapsto \hat{I} \otimes \hat{Z} \otimes \hat{Z}. \end{cases} \quad (6.84)$$

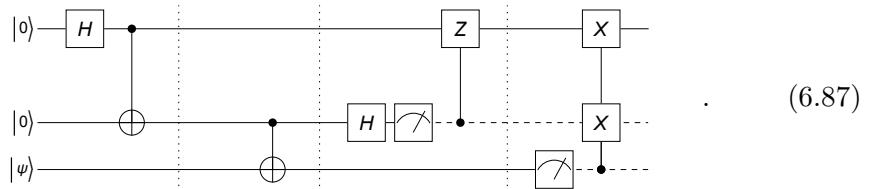
Next, we measure the observable \hat{X} on the qubit B , $\hat{M} \equiv \hat{I} \otimes \hat{X} \otimes \hat{I}$. Note that \hat{M} anti-commutes with the generator \hat{G}_2 while it commutes with \hat{G}_1 . We thus have to apply \hat{G} conditioned on the measurement outcome. We update \hat{G}_2 by replacing it with \hat{M} . It also anti-commutes with the logical operator \bar{Z} whereas it commutes with \bar{X} . We need to replace the logical operator \bar{Z} by $\hat{G}_2 \bar{Z}$. The two replacements leads to new generators and logical operators as

$$\begin{cases} \hat{G}_1 \mapsto \hat{X} \otimes \hat{X} \otimes \hat{X}, \\ \hat{G}_2 \mapsto \hat{I} \otimes \hat{X} \otimes \hat{I}; \end{cases} \quad \begin{cases} \bar{X} \mapsto \hat{I} \otimes \hat{I} \otimes \hat{X}, \\ \bar{Z} \mapsto \hat{Z} \otimes \hat{I} \otimes \hat{Z}. \end{cases} \quad (6.85)$$

At this stage, it is safe to drop the qubit B off since its state does not change nor is used after the measurement outcome is recorded. Finally, we measure the observable \hat{Z} on the qubit C , $\hat{M} \equiv \hat{I} \otimes \hat{I} \otimes \hat{Z}$. We note that \hat{M} anti-commutes with the generator \hat{G}_1 and the logical operator \bar{X} . In accordance with the same prescription as above, we apply \hat{G}_1 conditioned on the measurement outcome, and the generators and the logical operators are updated as follows

$$\begin{cases} \hat{G}_1 \mapsto \hat{I} \otimes \hat{I} \otimes \hat{Z}, \\ \hat{G}_2 \mapsto \hat{I} \otimes \hat{X} \otimes \hat{I}; \end{cases} \quad \begin{cases} \bar{X} \mapsto \hat{X} \otimes \hat{X} \otimes \hat{I}, \\ \bar{Z} \mapsto \hat{Z} \otimes \hat{I} \otimes \hat{Z}. \end{cases} \quad (6.86)$$

A quantum circuit model that follows the transformations in (6.85) and (6.86) more closely would look like the following



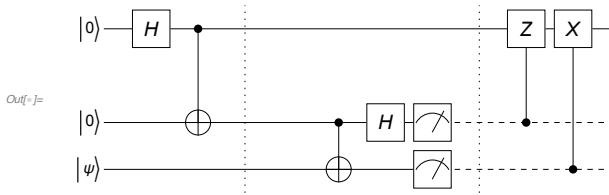
However, the operation \hat{X} on the second qubit is meaningless as the qubit was already measured. Therefore, the quantum circuit model is equivalent to the standard model for quantum teleportation.

We discuss again quantum teleportation, this time, in the stabilizer formalism.

```
In[5]:= Let[Complex, c]
in = ProductState[S[3] → c@{0, 1}, "Label" → Ket[ψ]]
Out[5]= (c₀ | 0⟩ + c₁ | 1⟩) S₃
```

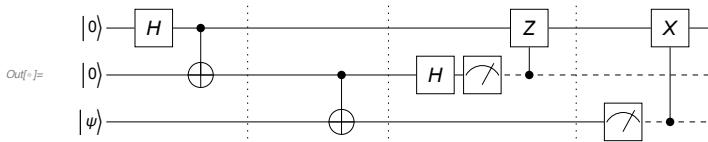
This is a quantum circuit model for quantum teleportation. It consists of three stages separated by the vertical dashed lines in the quantum circuit model. The first stage is to share the entangled pair of qubits. The second in the middle corresponds to the Bell measurement. The final stage transmits the measurement outcomes to Alice, and Alice adjusts the state of her qubit according to the classical information.

```
In[6]:= qc = QuantumCircuit[in, LogicalForm[Ket[], S@{1, 2}], S[1, 6],
CNOT[S[1], S[2]], "Separator", "Spacer", CNOT[S[2], S[3]],
S[2, 6], {Measurement[S[2]], Measurement[S[3]]}, "Separator",
ControlledU[S[2], S[1, 3]], ControlledU[S[3], S[1, 1]], "Invisible" → S[1.5]]
```



This rearranges the quantum circuit elements to re-analyze the quantum circuit model in stabilizer formalism.

```
In[7]:= qc2 = QuantumCircuit[
in, LogicalForm[Ket[], S@{1, 2}],
S[1, 6], CNOT[S[1], S[2]], "Separator", "Spacer",
CNOT[S[2], S[3]], "Separator",
S[2, 6], Measurement[S[2]], ControlledU[S[2], S[1, 3]], "Separator",
Measurement[S[3]], ControlledU[S[3], S[1, 1]]]
```



Now we test the result by performing the quantum circuit model many times.

```
In[8]:= out = Table[ExpressionFor[qc2], {4}] /.
c[0] * Conjugate[c[0]] + c[1] * Conjugate[c[1]] → 1;
LogicalForm@QuissoFactor[#, S@{2, 3}] & /@ out // TableForm
```

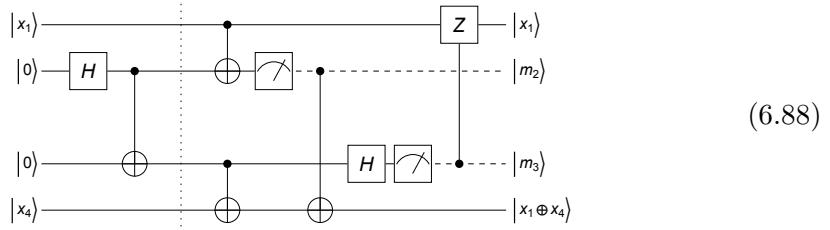
```
Out[8]//TableForm=
| 1s₂ 0s₃⟩ ⊗ (c₀ | 0s₁⟩ + c₁ | 1s₁⟩)
| 0s₂ 0s₃⟩ ⊗ (c₀ | 0s₁⟩ + c₁ | 1s₁⟩)
| 0s₂ 0s₃⟩ ⊗ (c₀ | 0s₁⟩ + c₁ | 1s₁⟩)
| 0s₂ 0s₃⟩ ⊗ (c₀ | 0s₁⟩ + c₁ | 1s₁⟩)
```

Each time the final state on the first qubit is the same regardless of the outcome of the measurements on the second and third qubit.

```
In[=]:= out = Table[ExpressionFor[qc2], {4}] /.
  c[0] * Conjugate[c[0]] + c[1] * Conjugate[c[1]] -> 1;
  LogicalForm@QuissoFactor[#, S@{2, 3}] & /@out // TableForm
Out[=]/TableForm=
|0_{S_2}0_{S_3}\rangle\otimes(c_0|0_{S_1}\rangle+c_1|1_{S_1}\rangle)
|0_{S_2}1_{S_3}\rangle\otimes(c_0|0_{S_1}\rangle+c_1|1_{S_1}\rangle)
|1_{S_2}1_{S_3}\rangle\otimes(c_0|0_{S_1}\rangle+c_1|1_{S_1}\rangle)
|0_{S_2}0_{S_3}\rangle\otimes(c_0|0_{S_1}\rangle+c_1|1_{S_1}\rangle)

In[=]:= out = Table[ExpressionFor[qc2], {4}] /.
  c[0] * Conjugate[c[0]] + c[1] * Conjugate[c[1]] -> 1;
  LogicalForm@QuissoFactor[#, S@{2, 3}] & /@out // TableForm
Out[=]/TableForm=
|0_{S_2}0_{S_3}\rangle\otimes(c_0|0_{S_1}\rangle+c_1|1_{S_1}\rangle)
|0_{S_2}0_{S_3}\rangle\otimes(c_0|0_{S_1}\rangle+c_1|1_{S_1}\rangle)
|0_{S_2}0_{S_3}\rangle\otimes(c_0|0_{S_1}\rangle+c_1|1_{S_1}\rangle)
|1_{S_2}1_{S_3}\rangle\otimes(c_0|0_{S_1}\rangle+c_1|1_{S_1}\rangle)
```

The second example is a remote implementation of CNOT gate using the pre-shared entangled pair of qubits: Suppose that Alice and Bob are located far away from each other, and there is no quantum channel available between them. Fortunately, they had shared and are maintaining a maximally entangled pair of qubits, and there is a classical channel available for communication between them. Alice want to implement a CNOT gate targeted on Bob's qubit—not the one involved in the shared entangled pair—controlled by her own qubit. The protocol is summarized in the following quantum circuit model.



The initial stage marked by the vertical dashed line is just for sharing an entangled pair, we start the analysis from this point. Then, the generators of the stabilizer are given by

$$\begin{cases} \hat{G}_1 = \hat{I} \otimes \hat{X} \otimes \hat{X} \otimes \hat{I}, \\ \hat{G}_2 = \hat{I} \otimes \hat{Z} \otimes \hat{Z} \otimes \hat{I}. \end{cases} \quad (6.89)$$

We also pick four logical operators to describe two *logical* qubits, one on Alcie's side (*A*) and the other on Bob's side (*B*),

$$\begin{cases} \bar{X}_A = \hat{X} \otimes \hat{I} \otimes \hat{I} \otimes \hat{I}, \\ \bar{Z}_A = \hat{Z} \otimes \hat{I} \otimes \hat{I} \otimes \hat{I}, \\ \bar{X}_B = \hat{I} \otimes \hat{I} \otimes \hat{I} \otimes \hat{X}, \\ \bar{Z}_B = \hat{I} \otimes \hat{I} \otimes \hat{I} \otimes \hat{Z}. \end{cases} \quad (6.90)$$

The subsequent two CNOT gates transforms the generators as

$$\begin{cases} \hat{G}_1 = \hat{I} \otimes \hat{X} \otimes \hat{X} \otimes \hat{X}, \\ \hat{G}_2 = \hat{Z} \otimes \hat{Z} \otimes \hat{Z} \otimes \hat{I}, \end{cases} \quad (6.91)$$

and the logical operators as

$$\begin{cases} \bar{X}_A = \hat{X} \otimes \hat{X} \otimes \hat{I} \otimes \hat{I}, \\ \bar{Z}_A = \hat{Z} \otimes \hat{I} \otimes \hat{I} \otimes \hat{I}, \\ \bar{X}_B = \hat{I} \otimes \hat{I} \otimes \hat{I} \otimes \hat{X}, \\ \bar{Z}_B = \hat{I} \otimes \hat{I} \otimes \hat{Z} \otimes \hat{Z}. \end{cases} \quad (6.92)$$

Next we measure \hat{Z} on the second qubit, $\hat{M} = \hat{I} \otimes \hat{Z} \otimes \hat{I} \otimes \hat{I}$. \hat{M} anti-commutes with \hat{G}_1 and \bar{X}_A . Therefore, we apply \hat{G}_1 conditioned on the measurement outcome, and update the generators and logical operators as

$$\begin{cases} \hat{G}_1 = \hat{I} \otimes \hat{Z} \otimes \hat{I} \otimes \hat{I}, \\ \hat{G}_2 = \hat{Z} \otimes \hat{Z} \otimes \hat{Z} \otimes \hat{I}, \end{cases} \quad (6.93)$$

and

$$\begin{cases} \bar{X}_A = \hat{X} \otimes \hat{I} \otimes \hat{X} \otimes \hat{X}, \\ \bar{Z}_A = \hat{Z} \otimes \hat{I} \otimes \hat{I} \otimes \hat{I}, \\ \bar{X}_B = \hat{I} \otimes \hat{I} \otimes \hat{I} \otimes \hat{X}, \\ \bar{Z}_B = \hat{I} \otimes \hat{I} \otimes \hat{Z} \otimes \hat{Z}. \end{cases} \quad (6.94)$$

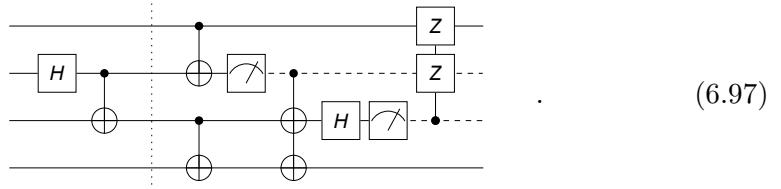
Finally, we measure \hat{X} on the third qubit, $\hat{M} = \hat{I} \otimes \hat{I} \otimes \hat{X} \otimes \hat{I}$. \hat{M} anti-commutes with \hat{G}_2 and \bar{Z}_B . We apply \hat{G}_2 conditioned on the measurement outcome. The generators and logical operators are transformed as

$$\begin{cases} \hat{G}_1 = \hat{I} \otimes \hat{Z} \otimes \hat{I} \otimes \hat{I}, \\ \hat{G}_2 = \hat{I} \otimes \hat{I} \otimes \hat{X} \otimes \hat{I}, \end{cases} \quad (6.95)$$

and

$$\begin{cases} \bar{X}_A = \hat{X} \otimes \hat{I} \otimes \hat{X} \otimes \hat{X}, \\ \bar{Z}_A = \hat{Z} \otimes \hat{I} \otimes \hat{I} \otimes \hat{I}, \\ \bar{X}_B = \hat{I} \otimes \hat{I} \otimes \hat{I} \otimes \hat{X}, \\ \bar{Z}_B = \hat{Z} \otimes \hat{Z} \otimes \hat{I} \otimes \hat{Z}. \end{cases} \quad (6.96)$$

At this stage, the generators are meaningless since the second and third qubits are now classical after the two measurements. If one followed the above steps of analysis literally, she would get a quantum circuit model of the form



However, the classically-conditioned operation of \hat{X} on the third qubit right before the measurement of \hat{X} does not affect the result. On the other hand, the semiclassically-conditioned operation of \hat{Z} on the second qubit is meaningless because the qubit was already measured.

Here we investigate the procedure to remotely implement a CNOT gate.

This is the quantum circuit model we want to analyze.

```
qc = QuantumCircuit[S[2, 6], CNOT[S[2], S[3]], "Separator", "Spacer",
  {CNOT[S[1], S[2]], CNOT[S[3], S[4]]},
  Measurement[S[2]], CNOT[S[2], S@{3, 4}],
  S[3, 6], Measurement[S[3]], ControlledU[S[3], S[1, 3]]];
```

This is the general form of the input state.

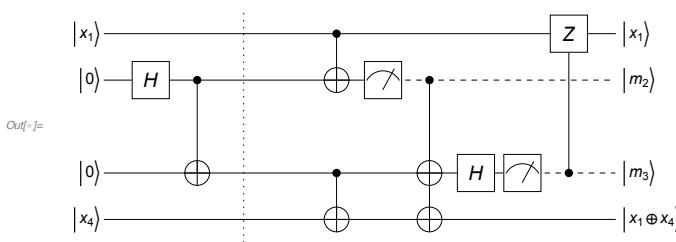
```
Let[Integer, x, m]
in = LogicalForm[Ket[S[1] → x[1], S[4] → x[4]], S@{1, 2, 3, 4}];
```

This is the expected form of the output state.

```
out = Ket[S[1] → x[1], S@{2, 3} → m@{2, 3}, S[4] → Mod[x[1] + x[4], 2]];
```

This shows the quantum circuit model with the expected output states specified.

```
In[7]:= qc1 = QuantumCircuit[in, qc, out, "PortSize" → {0.65, 2}, "Invisible" → S[2.5]]
```



Check whether the quantum circuit model works as expected.

```
In[8]:= bs = Basis[S@{1, 4}];
new = qc ** bs;
Thread[LogicalForm[bs] → LogicalForm[new, S@{1, 2, 3, 4}]] // TableForm
Out[8]//TableForm=
| 0s1 0s4 > → | 0s1 0s2 1s3 0s4 >
| 0s1 1s4 > → | 0s1 0s2 0s3 1s4 >
| 1s1 0s4 > → | 1s1 1s2 0s3 1s4 >
| 1s1 1s4 > → | 1s1 1s2 0s3 0s4 >
```

6.4 Stabilizer Codes

Stabilizer codes is a class of quantum codes the construction of which is based on the stabilizer formalism. It was put forward by Gottesman (Gottesman, 1996, 1998).

How does the stabilizer construction work? Let \mathcal{S} be the stabilizer of a code space \mathcal{V} . Suppose that \mathcal{V} is corrupted by an error operator \hat{E} in the Pauli group $\mathcal{P}(n)$. What would happen to the code space? There are three distinctive cases:^{6.6} First, \hat{E} anti-commutes with one^{6.7} of the generators, say \hat{G} , of the stabilizer \mathcal{S} . In this case, \hat{E} takes the code space \mathcal{V} to a subspace orthogonal to \mathcal{V} . To see this, suppose that $|\beta\rangle$ is a vector in the code space, and $|\beta'\rangle := \hat{E}|\beta\rangle$ is its corrupted state. Note that $\hat{G}\hat{E} + \hat{E}\hat{G} = 0$, and hence $\langle\alpha|\hat{G}\hat{E}|\beta\rangle + \langle\alpha|\hat{E}\hat{G}|\beta\rangle = 0$ for all $|\alpha\rangle$ in the code space. As both $|\alpha\rangle$ and $|\beta\rangle$ are stabilized by \hat{G} , we observe that $\langle\alpha|\beta'\rangle = 0$. Therefore, in this case, one can detect the error, and correct it. Second, \hat{E} belongs to the stabilizer \mathcal{S} . In this case, \hat{E} itself stabilizes the code space, and the code space remains intact without any corruption. Third, \hat{E} commutes with all elements of the stabilizer, but does not belong to the stabilizer. This is the most dangerous case. As we have seen in Section 6.3.5, measuring \hat{E} can reveal some information of the quantum states in the code space. It implies that \hat{E} breaks the code space, and the corruption cannot be corrected.

The above observation already suggests that for an error to be correctable, it needs to either anti-commute with a generator of the stabilizer or belong to the stabilizer. However, usually there are a set of errors and we want to protect quantum states against combinations of them. Hence the error-correction conditions for stabilizer codes are a bit more involved. Nevertheless, the error-correction conditions in Section 6.2.1 translate more transparently in stabilizer formalism: Let \mathcal{S} be the stabilizer of a code space \mathcal{V} . Suppose that the errors are described by a set $\{\hat{E}_\mu\}$ of operators in the Pauli group $\mathcal{P}(n)$. The necessary and sufficient condition for the code to correct the errors is that for all μ and ν , $\hat{E}_\mu^\dagger \hat{E}_\nu$ either belongs to \mathcal{S} or anti-commutes with at least one generator of \mathcal{S} . Remember that the identity operator is always one of the error operators and thus the condition include the above observation.

How do we actually perform the recovery operation to correct the error? Let $\{\hat{G}_1, \dots, \hat{G}_k\}$ be a set of independent generators of the stabilizer \mathcal{S} . Suppose that $\{\hat{E}_\mu\}$ is the set of error operators *correctable* by the code. The first step is to detect error syndrome. It is achieved by measuring the generators $\hat{G}_1, \dots, \hat{G}_k$. Each outcome ± 1 serves as a symptom of errors. If an error \hat{E}_μ occurs, then it causes a syndrome $(\varepsilon_1^\mu, \dots, \varepsilon_k^\mu)$. The error and syndrome are related by

$$\hat{E}_\mu \hat{G}_j \hat{E}_\mu^\dagger = \varepsilon_j^\mu \hat{G}_j. \quad (6.98)$$

^{6.6}The same classification of relationships between an operator and a given stabilizer was used to analyze the effects of measurements in Section 6.3.5.

^{6.7}Without loss of generality, we can assume that \hat{E} anti-commutes with only one of the generator.

When a syndrome is associated with a single error \hat{E}_μ , we can correct it simply by applying \hat{E}_μ^\dagger . What if two different errors \hat{E}_μ and \hat{E}_ν cause the same syndrome? In other words, suppose that

$$(\varepsilon_1^\mu, \dots, \varepsilon_k^\mu) = (\varepsilon_1^\nu, \dots, \varepsilon_k^\nu). \quad (6.99)$$

In this case, we have

$$\hat{E}_\mu \hat{P} \hat{E}_\mu^\dagger = \hat{E}_\nu \hat{P} \hat{E}_\nu^\dagger, \quad (6.100)$$

whence

$$(\hat{E}_\mu^\dagger \hat{E}_\nu) \hat{P} (\hat{E}_\mu^\dagger \hat{E}_\nu)^\dagger = \hat{P}, \quad (6.101)$$

where \hat{P} is the projection onto the code space. It means that $\hat{E}_\mu^\dagger \hat{E}_\nu$ is a member of the stabilizer. That is, even if it was the error \hat{E}_ν that actually occurred, it can still be corrected by applying \hat{E}_μ^\dagger . Therefore, however many errors are associated with a given syndrome, one can choose any error \hat{E}_μ among them, and apply \hat{E}_μ^\dagger to recover the original state.

One remaining task for a given stabilizer code is the encoding procedure. How can one prepare the system of n physical qubits, say, in the logical basis state $|\bar{0} \dots \bar{0}\rangle$? It requires a procedure without referring to an explicit form of the logical basis states. One of the simplest approach is to measure the generators $\hat{G}_1, \dots, \hat{G}_k$ and the logical operators $\hat{Z}_1, \dots, \hat{Z}_{n-k}$.^{6.8} Recall that the combined set $\{\hat{G}_1, \dots, \hat{G}_k, \hat{Z}_1, \dots, \hat{Z}_{n-k}\}$ consists of all mutually commuting and independent operators. It thus generates a stabilizer subgroup stabilizing a one-dimensional subspace, which is spanned by nothing but the desired state $|\bar{0} \dots \bar{0}\rangle$. Therefore, one can regard the measurement of the generators and logical operators as the error syndrome detection. Then, the initialization to the desired state corresponds to a recovery procedure discussed above.

Let us take the encoding procedure in more explicit steps: Each measurement of $\hat{G}_1, \dots, \hat{G}_k$ and $\hat{Z}_1, \dots, \hat{Z}_{n-k}$ yields measurement outcome ± 1 . Depending on the measurement results, the final state $|\psi\rangle$ after all measurements is stabilized by $\pm \hat{G}_1, \dots, \pm \hat{G}_k, \pm \hat{Z}_1, \dots, \pm \hat{Z}_{n-k}$. We then recall—see Section 6.3.1—that there exist a Pauli operator \hat{G} that anti-commutes with one and only one of the independent operators but commutes with all the others. We find such a Pauli operator for each observable with measurement outcome -1 , and apply those Pauli operators on $|\psi\rangle$. Once the state $|\bar{0} \dots \bar{0}\rangle$ is achieved, other logical basis states can be obtained by applying the logical operators $\hat{X}_1, \dots, \hat{X}_{n-k}$ as necessary.

In the remainder of the section, we take some examples of stabilizer codes. They were constructed in a different scheme, but here reconstruct them in the stabilizer formalism.

^{6.8} Encoding and decoding a general stabilizer code using unitary gates are also possible and explained in detail in Cleve & Gottesman (1997).

$\hat{Z}_1\hat{Z}_2$	$\hat{Z}_2\hat{Z}_3$		error
1	1		\hat{I}
1	-1		\hat{X}_3
-1	1		\hat{X}_1
-1	-1		\hat{X}_2

Table 6.1: The error syndrome of the bit-flip correction code.

6.4.1 Bit-Flip Code

First consider the three-qubit bit-flip correction code. The code space is spanned by the two logical basis states $|0\rangle := |000\rangle$ and $|1\rangle := |111\rangle$. The stabilizer is generated by $\hat{Z}_1\hat{Z}_2$ and $\hat{Z}_2\hat{Z}_3$. Here \hat{X}_j , \hat{Y}_j , and \hat{Z}_j denote the Pauli operators acting on the qubit j . In this code, the logical operators are given by $\bar{Z} = \hat{Z}_1\hat{Z}_2\hat{Z}_3$ and $\bar{X} = \hat{X}_1\hat{X}_2\hat{X}_3$.

The code can correct the bit-flip errors in $\{\hat{I}, \hat{X}_1, \hat{X}_2, \hat{X}_3\}$. The error syndrome is detected by measuring the generators $\hat{Z}_1\hat{Z}_2$ and $\hat{Z}_2\hat{Z}_3$. Table 6.1 shows the error syndromes for bit-flip errors on different qubits.

It is heuristic to consider an error described by \hat{Z}_1 . It commutes with both generators $\hat{Z}_1\hat{Z}_2$ and $\hat{Z}_2\hat{Z}_3$ but is not an element of the stabilizer. It means that the code is not protected against the error \hat{Z}_1 . It is also the case with \hat{Z}_2 and \hat{Z}_3 . It is not surprising because these operators correspond to phase-flip errors. The code is designed only for bit-flip errors and does not protect against phase-flip errors.

Here are the generators of the stabilizer of the code space for the bit-flip correction code.

```
In[7]:= gnr = {S[1, 3] ** S[2, 3], S[2, 3] ** S[3, 3]};
gnr // PauliForm
Out[7]= {Z ⊗ Z ⊗ I, I ⊗ Z ⊗ Z}
```

These are error operators correctable by the code.

```
In[8]:= err = {I, S[1, 1], S[2, 1], S[3, 1]};
err // PauliForm
Out[8]= {I ⊗ I ⊗ I, X ⊗ I ⊗ I, I ⊗ X ⊗ I, I ⊗ I ⊗ X}

In[9]:= chk = Union[Multiply @@@ Choices[err, {2}]];
chk // PauliForm
Out[9]= {I ⊗ I ⊗ I, X ⊗ X ⊗ I, X ⊗ I ⊗ X, I ⊗ X ⊗ X, X ⊗ I ⊗ I, I ⊗ X ⊗ I, I ⊗ I ⊗ X}
```

Check and confirm that the errors are indeed correctable. To do it, we use `GottesmanTest`.

$\hat{X}_1\hat{X}_2$	$\hat{X}_2\hat{X}_3$	error
1	1	\hat{I}
1	-1	\hat{Z}_3
-1	1	\hat{Z}_1
-1	-1	\hat{Z}_2

Table 6.2: The error syndrome of the phase-flip correction code.

In[4]:= ?GottesmanTest

Symbol

GottesmanTest [a, b] returns 1 if the two operators a and b commute with each other, -1 if they anti-commute, and 0 otherwise.

▼

As one can see, there is only one combination, $\mathbf{I} \otimes \mathbf{I} \otimes \mathbf{I}$, of the error operators that commutes with all generators of the stabilizer. But it belongs to the stabilizer.

In[5]:= mat = Outer[GottesmanTest, gnr, chk];
TableForm[mat, TableHeadings → {PauliForm@gnr, PauliForm@chk},
TableAlignments → Right]

	$\mathbf{I} \otimes \mathbf{I} \otimes \mathbf{I}$	$\mathbf{X} \otimes \mathbf{X} \otimes \mathbf{I}$	$\mathbf{X} \otimes \mathbf{I} \otimes \mathbf{X}$	$\mathbf{I} \otimes \mathbf{X} \otimes \mathbf{X}$	$\mathbf{X} \otimes \mathbf{I} \otimes \mathbf{I}$	$\mathbf{I} \otimes \mathbf{X} \otimes \mathbf{I}$	$\mathbf{I} \otimes \mathbf{I} \otimes \mathbf{X}$
$\mathbf{Z} \otimes \mathbf{Z} \otimes \mathbf{I}$	1	1	-1	-1	-1	-1	1
$\mathbf{I} \otimes \mathbf{Z} \otimes \mathbf{Z}$	1	-1	-1	1	1	-1	-1

This table displays the error syndrome of the bit-flip correction code.

In[6]:= syndrome = Map[(# ** gnr ** #) / gnr &, err];
TableForm[syndrome, TableAlignments → Right,
TableHeadings → {PauliForm@err, PauliForm@gnr}]

	$\mathbf{Z} \otimes \mathbf{Z} \otimes \mathbf{I}$	$\mathbf{I} \otimes \mathbf{Z} \otimes \mathbf{Z}$
$\mathbf{I} \otimes \mathbf{I} \otimes \mathbf{I}$	1	1
$\mathbf{X} \otimes \mathbf{I} \otimes \mathbf{I}$	-1	1
$\mathbf{I} \otimes \mathbf{X} \otimes \mathbf{I}$	-1	-1
$\mathbf{I} \otimes \mathbf{I} \otimes \mathbf{X}$	1	-1

6.4.2 Phase-Flip Code

Let us now turn to the three-qubit phase-flip correction code. The code space is spanned by the two logical basis states $|\bar{0}\rangle := |+++ \rangle$ and $|\bar{1}\rangle := |--- \rangle$, where $|\pm\rangle := (|0\rangle \pm |1\rangle)/\sqrt{2}$. The stabilizer is generated by $\hat{X}_1\hat{X}_2$ and $\hat{X}_2\hat{X}_3$. In this code, the logical operators are given by $\bar{Z} = \hat{X}_1\hat{X}_2\hat{X}_3$ and $\bar{X} = \hat{Z}_1\hat{Z}_2\hat{Z}_3$.

The code can correct the phase-flip errors in $\{\hat{I}, \hat{Z}_1, \hat{Z}_2, \hat{Z}_3\}$. The error syndrome is detected by measuring the generators $\hat{X}_1\hat{X}_2$ and $\hat{X}_2\hat{X}_3$. Table 6.2 shows the error syndromes for bit-flip errors on different qubits.

The above accounts are a direct translation of the phase-flip correction code as presented in Section 6.1.2. However, the bit-flip and phase-flip correction codes are

closely related to each other. A comparison of the error syndromes in Tables 6.1 and 6.2 of the two codes suggests it. One can make the relation clearer and more rigorous in stabilizer formalism: The code space of the phase-flip correction code is an image of a unitary transformation, more specifically, a Clifford operator $\hat{U} := \hat{H}_1 \hat{H}_2 \hat{H}_3$,

$$|+++ \rangle = \hat{U} |000\rangle, \quad |--- \rangle = \hat{U} |111\rangle. \quad (6.102)$$

In accordance with the transformation rules of stabilizers under unitary transformations discussed in Section 6.3.3, it follows that the generators of the phase-flip correction code are related to those of the bit-flip correction code by the same Clifford operator,

$$\hat{X}_1 \hat{X}_2 = \hat{U} \hat{Z}_1 \hat{Z}_2 \hat{U}^\dagger, \quad \hat{X}_2 \hat{X}_3 = \hat{U} \hat{Z}_2 \hat{Z}_3 \hat{U}^\dagger. \quad (6.103)$$

The error syndromes in Tables 6.1 and 6.2 are certainly related to each other by the same Clifford operator \hat{U} .

Here are the generators of the stabilizer of the code space for the bit-flip correction code.

```
In[7]:= gnr = {S[1, 1] ** S[2, 1], S[2, 1] ** S[3, 1]};
gnr // PauliForm
Out[7]= {X ⊗ X ⊗ I, I ⊗ X ⊗ X}
```

These are error operators correctable by the code.

```
In[8]:= err = {I, S[1, 3], S[2, 3], S[3, 3]};
err // PauliForm
Out[8]= {I ⊗ I ⊗ I, Z ⊗ I ⊗ I, I ⊗ Z ⊗ I, I ⊗ I ⊗ Z}
In[9]:= chk = Union[Multiply @@@ Choices[err, {2}]];
chk // PauliForm
Out[9]= {I ⊗ I ⊗ I, Z ⊗ Z ⊗ I, Z ⊗ I ⊗ Z, I ⊗ Z ⊗ Z, Z ⊗ I ⊗ I, I ⊗ Z ⊗ I, I ⊗ I ⊗ Z}
```

Check and confirm that the errors are indeed correctable. As one can see, there is only one combination, $I \otimes I \otimes I$, of the error operators that commutes with all generators of the stabilizer. But it belongs to the stabilizer.

```
In[10]:= mat = Outer[GottesmanTest, gnr, chk];
TableForm[mat, TableHeadings → {PauliForm@gnr, PauliForm@chk},
TableAlignments → Right]
Out[10]/TableForm=
```

	$I \otimes I \otimes I$	$Z \otimes Z \otimes I$	$Z \otimes I \otimes Z$	$I \otimes Z \otimes Z$	$Z \otimes I \otimes I$	$I \otimes Z \otimes I$	$I \otimes I \otimes Z$
$X \otimes X \otimes I$	1	1	-1	-1	-1	-1	1
$I \otimes X \otimes X$	1	-1	-1	1	1	-1	-1

This table displays the error syndrome of the phase-flip correction code.

```
In[11]:= syndrome = Map[(# ** gnr ** #) / gnr &, err];
TableForm[syndrome, TableAlignments → Right,
TableHeadings → {PauliForm@err, PauliForm@gnr}]
Out[11]/TableForm=
```

	$X \otimes X \otimes I$	$I \otimes X \otimes X$
$I \otimes I \otimes I$	1	1
$Z \otimes I \otimes I$	-1	1
$I \otimes Z \otimes I$	-1	-1
$I \otimes I \otimes Z$	1	-1

symbol	operator
\hat{G}_1	$\hat{Z} \otimes \hat{Z} \otimes \hat{I} \otimes \hat{I} \otimes \hat{I} \otimes \hat{I} \otimes \hat{I} \otimes \hat{I} \otimes \hat{I}$
\hat{G}_2	$\hat{I} \otimes \hat{Z} \otimes \hat{Z} \otimes \hat{I} \otimes \hat{I} \otimes \hat{I} \otimes \hat{I} \otimes \hat{I} \otimes \hat{I}$
\hat{G}_3	$\hat{I} \otimes \hat{I} \otimes \hat{I} \otimes \hat{Z} \otimes \hat{Z} \otimes \hat{I} \otimes \hat{I} \otimes \hat{I} \otimes \hat{I}$
\hat{G}_4	$\hat{I} \otimes \hat{I} \otimes \hat{I} \otimes \hat{I} \otimes \hat{Z} \otimes \hat{Z} \otimes \hat{I} \otimes \hat{I} \otimes \hat{I}$
\hat{G}_5	$\hat{I} \otimes \hat{I} \otimes \hat{I} \otimes \hat{I} \otimes \hat{I} \otimes \hat{I} \otimes \hat{Z} \otimes \hat{Z} \otimes \hat{I}$
\hat{G}_6	$\hat{I} \otimes \hat{I} \otimes \hat{I} \otimes \hat{I} \otimes \hat{I} \otimes \hat{I} \otimes \hat{I} \otimes \hat{Z} \otimes \hat{Z}$
\hat{G}_7	$\hat{X} \otimes \hat{X} \otimes \hat{X} \otimes \hat{X} \otimes \hat{X} \otimes \hat{X} \otimes \hat{I} \otimes \hat{I} \otimes \hat{I}$
\hat{G}_8	$\hat{I} \otimes \hat{I} \otimes \hat{I} \otimes \hat{X} \otimes \hat{X} \otimes \hat{X} \otimes \hat{X} \otimes \hat{X} \otimes \hat{X}$

Table 6.3: A set of independent generators of the stabilizer for Shor’s nine-qubit code.

6.4.3 Nine-Qubit Code

Let us now combine the bit-flip and phase-flip correction codes to get Shor’s nine-qubit code that corrects arbitrary single-qubit errors. A set of independent generators of the stabilizer of the code is listed in Table 6.3. The logical operators are given by

$$\begin{aligned}\bar{Z} &= \hat{X} \otimes \hat{X}, \\ \bar{X} &= \hat{Z} \otimes \hat{Z}.\end{aligned}\quad (6.104)$$

The errors are described by the set of single-qubit Pauli operators

$$\{\hat{I}_j, \hat{X}_j, \hat{Y}_j, \hat{Z}_j : j = 1, 2, \dots, 9\}. \quad (6.105)$$

It is rather tedious and yet straightforward to check by inspection that the above set of errors satisfy the error-correction conditions discussed at the beginning of the section.

Shor’s code needs 9 qubits to encode physical single-qubit quantum states.

jj = Range[9];

This is a generating set of the stabilizer of the code.

```
In[7]:= gnr = {
  S[1, 3] ** S[2, 3], S[2, 3] ** S[3, 3],
  S[4, 3] ** S[5, 3], S[5, 3] ** S[6, 3],
  S[7, 3] ** S[8, 3], S[8, 3] ** S[9, 3],
  Multiply@@S[{1, 2, 3, 4, 5, 6}, 1],
  Multiply@@S[{4, 5, 6, 7, 8, 9}, 1]}
```

```
Out[7]= {S1z S2z, S2z S3z, S4z S5z, S5z S6z, S7z S8z, S8z S9z, S1x S2x S3x S4x S5x S6x, S4x S5x S6x S7x S8x S9x}
```

These are logical operators of the code.

```
In[7]:= opX = Multiply @@ S[jj, 1]
```

```
opZ = Multiply @@ S[jj, 3]
```

```
Out[7]= S1X S2X S3X S4X S5X S6X S7X S8X S9X
```

```
Out[7]= S1Z S2Z S3Z S4Z S5Z S6Z S7Z S8Z S9Z
```

These are the error operators correctable by the code.

```
In[8]:= err = Prepend[S[jj, All], 1]
```

```
Out[8]= {1, S1X, S1Y, S1Z, S2X, S2Y, S2Z, S3X, S3Y, S3Z, S4X, S4Y, S4Z, S5X, S5Y, S5Z, S6X, S6Y, S6Z, S7X, S7Y, S7Z, S8X, S8Y, S8Z, S9X, S9Y, S9Z}
```

In order to check that the above errors are indeed correctable, we consider combinations of two error operators.

```
In[9]:= chk = Union[Multiply @@ Choices[err, {2}]];
Length[chk]
```

```
Out[9]= 379
```

This calculates the commutation of the above combinations with the generators of the stabilizer.

```
In[10]:= Timing[mat = Outer[GottesmanTest, chk, gnr];]
```

```
Out[10]= {9.9689, Null}
```

This displays a *part* of the commutation relation. One can see that each of them either commutes or anti-commutes with the generators.

```
In[11]:= TableForm[mat[[;; 5, ;; 5, 6]],
TableAlignments -> Right, TableHeadings -> {chk[[;; 5]], gnr}]
```

	S ₁ ^Z S ₂ ^Z	S ₂ ^Z S ₃ ^Z	S ₄ ^Z S ₅ ^Z	S ₅ ^Z S ₆ ^Z	S ₇ ^Z S ₈ ^Z	S ₈ ^Z S ₉ ^Z
1	1	1	1	1	1	1
S ₁ ^X S ₂ ^X	1	-1	1	1	1	1
S ₁ ^X S ₂ ^Y	1	-1	1	1	1	1
S ₁ ^X S ₂ ^Z	-1	1	1	1	1	1
S ₁ ^X S ₃ ^X	-1	-1	1	1	1	1

The most dangerous case is that the check operator commutes with all of the generators but does not belong to the stabilizer. To check if there is such case, we examine the check operators that commute with all of the generators.

```
In[12]:= kk = Catenate@MapIndexed[If[ContainsOnly[#, {1}], #2, Nothing] &, mat]
```

```
Out[12]= {1, 52, 55, 118, 223, 226, 262, 313, 316, 325}
```

This shows that such check operators actually belong to the stabilizer. The considered errors are thus correctable by the code.

```
In[13]:= chk[[kk]]
```

```
Out[13]= {1, S1Z S2Z, S1Z S3Z, S2Z S3Z, S4Z S5Z, S4Z S6Z, S5Z S6Z, S7Z S8Z, S7Z S9Z, S8Z S9Z}
```

This table displays a *part* of the error syndrome of the code.

	S_1^z	S_2^z	S_3^z	S_4^z	S_5^z	S_6^z	S_7^z	S_8^z	S_1^x	S_2^x	S_3^x	S_4^x	S_5^x	S_6^x	S_7^x	S_8^x
1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1
S_1^y	-1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1
S_1^z	-1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1
S_2^y	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1
S_2^z	-1	-1	1	1	1	1	1	1	1	1	1	1	1	1	1	1
S_3^y	-1	-1	1	1	1	1	1	1	1	1	1	1	1	1	1	1
S_3^z	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1
S_4^y	1	-1	1	1	1	1	1	1	1	1	1	1	1	1	1	1
S_4^z	1	-1	1	1	1	1	1	1	1	1	1	1	1	1	1	1

6.4.4 Five-Qubit Code

The smallest possible code to protect single-qubit states against generic errors on single qubits is the five-qubit code discovered independently by [Bennett et al. \(1996\)](#) and by [Laflamme et al. \(1996\)](#). It belongs to a wider class of codes called *Calderbank-Shor-Steane (CSS) codes* ([Calderbank & Shor, 1996](#); [Steane, 1996](#)). CSS codes were initially developed in analogy with the classical linear codes. But one can also regard it as a special type of stabilizer codes.

The stabilizer of the five-qubit code is generated by the following four generators

$$\begin{aligned}\hat{G}_1 &= \hat{X} \otimes \hat{Z} \otimes \hat{Z} \otimes \hat{X} \otimes \hat{I}, \\ \hat{G}_2 &= \hat{I} \otimes \hat{X} \otimes \hat{Z} \otimes \hat{Z} \otimes \hat{X}, \\ \hat{G}_3 &= \hat{X} \otimes \hat{I} \otimes \hat{X} \otimes \hat{Z} \otimes \hat{Z}, \\ \hat{G}_4 &= \hat{Z} \otimes \hat{X} \otimes \hat{I} \otimes \hat{X} \otimes \hat{Z}.\end{aligned}\tag{6.106}$$

It can correct arbitrary errors on single qubits,

$$\{\hat{I}_j, \hat{X}_j, \hat{Y}_j, \hat{Z}_j : j = 1, 2, 3, 4, 5\},\tag{6.107}$$

which can be confirmed by checking the commutation relations of the combinations of errors in (6.107) with the generators in (6.106). One can choose

$$\bar{Z} = \hat{Z} \otimes \hat{Z} \otimes \hat{Z} \otimes \hat{Z} \otimes \hat{Z}, \quad \bar{X} = \hat{X} \otimes \hat{X} \otimes \hat{X} \otimes \hat{X} \otimes \hat{X}\tag{6.108}$$

as logical operators of the code.

We consider a code encoded on 5 qubits.

```
jj = Range[5];
```

Here are the generators of the stabilizer of the five-qubit code.

```
In[7]:= gnr = {
  S[1, 1] ** S[2, 3] ** S[3, 3] ** S[4, 1],
  S[2, 1] ** S[3, 3] ** S[4, 3] ** S[5, 1],
  S[1, 1] ** S[3, 1] ** S[4, 3] ** S[5, 3],
  S[1, 3] ** S[2, 1] ** S[4, 1] ** S[5, 3]};
PauliForm[gnr]
Out[7]= {X ⊗ Z ⊗ Z ⊗ X ⊗ I, I ⊗ X ⊗ Z ⊗ Z ⊗ X, X ⊗ I ⊗ X ⊗ Z ⊗ Z, Z ⊗ X ⊗ I ⊗ X ⊗ Z}
```

These are logical operators of the code.

```
In[5]:= opX = Multiply @@ S[jj, 1];
PauliForm[opX]
opZ = Multiply @@ S[jj, 3];
PauliForm[opZ]

Out[5]= X ⊗ X ⊗ X ⊗ X ⊗ X

Out[6]= Z ⊗ Z ⊗ Z ⊗ Z ⊗ Z
```

Here are the error operators correctable by the code.

```
In[7]:= err = Prepend[S[jj, All], 1];
PauliForm[err]

Out[7]= {I ⊗ I ⊗ I ⊗ I ⊗ I, X ⊗ I ⊗ I ⊗ I ⊗ I, Y ⊗ I ⊗ I ⊗ I ⊗ I, Z ⊗ I ⊗ I ⊗ I ⊗ I,
I ⊗ X ⊗ I ⊗ I ⊗ I, I ⊗ Y ⊗ I ⊗ I ⊗ I, I ⊗ Z ⊗ I ⊗ I ⊗ I, I ⊗ I ⊗ X ⊗ I ⊗ I,
I ⊗ I ⊗ Y ⊗ I ⊗ I, I ⊗ I ⊗ Z ⊗ I ⊗ I, I ⊗ I ⊗ I ⊗ X ⊗ I, I ⊗ I ⊗ I ⊗ Y ⊗ I,
I ⊗ I ⊗ I ⊗ Z ⊗ I, I ⊗ I ⊗ I ⊗ I ⊗ X, I ⊗ I ⊗ I ⊗ I ⊗ Y, I ⊗ I ⊗ I ⊗ I ⊗ Z}

In[8]:= chk = Union[Multiply @@ Choices[err, {2}]];
Length[chk]

Out[8]= 121
```

```
In[9]:= Timing[mat = Outer[GottesmanTest, chk, gnr];]
Out[9]= {2.09133, Null}
```

This displays a *part* of the commutation relations of some of the check operators. One can see that each of them either commutes or anti-commutes with the generators.

```
In[10]:= TableForm[mat[[;; 5]], TableAlignments → Right,
TableHeadings → PauliForm@{chk[[;; 5]], gnr}]

Out[10]//TableForm=
```

	$X \otimes Z \otimes Z \otimes X \otimes I$	$I \otimes X \otimes Z \otimes Z \otimes X$	$X \otimes I \otimes X \otimes Z \otimes Z$	$Z \otimes X \otimes I \otimes X \otimes Z$
$I \otimes I \otimes I \otimes I \otimes I$	1	1	1	1
$X \otimes X \otimes I \otimes I \otimes I$	-1	1	1	-1
$X \otimes Y \otimes I \otimes I \otimes I$	-1	-1	1	1
$X \otimes Z \otimes I \otimes I \otimes I$	1	-1	1	1
$X \otimes I \otimes X \otimes I \otimes I$	-1	-1	1	-1

The most dangerous case is that the check operator commutes with all of the generators but does not belong to the stabilizer. To check if there is such case, we examine the check operators that commute with all of the generators.

```
In[11]:= kk = Catenate@MapIndexed[If[ContainsOnly[{#1, {1}}, #2, Nothing] &, mat]
Out[11]= {1}
```

This shows that such check operators actually belong to the stabilizer. The considered errors are thus correctable by the code.

```
In[12]:= chk[[kk]]
Out[12]= {1}
```

This shows the error syndrome of the code upon the measurement of the generators of the stabilizer.

	$X \otimes Z \otimes Z \otimes X \otimes I$	$I \otimes X \otimes Z \otimes Z \otimes X$	$X \otimes I \otimes X \otimes Z \otimes Z$	$Z \otimes X \otimes I \otimes X \otimes Z$
$I \otimes I \otimes I \otimes I \otimes I$	1	1	1	1
$X \otimes I \otimes I \otimes I \otimes I$	1	1	1	-1
$Y \otimes I \otimes I \otimes I \otimes I$	-1	1	-1	-1
$Z \otimes I \otimes I \otimes I \otimes I$	-1	1	-1	1
$I \otimes X \otimes I \otimes I \otimes I$	-1	1	1	1
$I \otimes Y \otimes I \otimes I \otimes I$	-1	-1	1	-1
$I \otimes Z \otimes I \otimes I \otimes I$	1	-1	1	-1
$I \otimes I \otimes X \otimes I \otimes I$	-1	-1	1	1
$I \otimes I \otimes Y \otimes I \otimes I$	-1	-1	-1	1
$I \otimes I \otimes Z \otimes I \otimes I$	1	1	-1	1
$I \otimes I \otimes I \otimes X \otimes I$	1	-1	-1	1
$I \otimes I \otimes I \otimes Y \otimes I$	-1	-1	-1	-1
$I \otimes I \otimes I \otimes Z \otimes I$	-1	1	1	-1
$I \otimes I \otimes I \otimes I \otimes X$	1	1	-1	-1
$I \otimes I \otimes I \otimes I \otimes Y$	1	-1	-1	-1
$I \otimes I \otimes I \otimes I \otimes Z$	1	-1	1	1

As mentioned before, in stabilizer formalism, any meaningful procedure can be performed without referring to logical basis states. Nevertheless, it is heuristically interesting to evaluate the logical basis states explicitly. They can be found by simultaneously diagonalizing the generators of the stabilizer and the logical operator \bar{Z} . For the five-qubit code, they are given (unnormalized) by

$$\begin{aligned} |\bar{0}\rangle = & |00000\rangle - |00011\rangle + |00101\rangle - |00110\rangle + |01001\rangle + |01010\rangle \\ & - |01100\rangle - |01111\rangle - |10001\rangle + |10010\rangle + |10100\rangle - |10111\rangle \\ & - |11000\rangle - |11011\rangle - |11101\rangle - |11110\rangle \quad (6.109) \end{aligned}$$

and

$$\begin{aligned} |\bar{1}\rangle = & -|00001\rangle - |00010\rangle - |00100\rangle - |00111\rangle - |01000\rangle + |01011\rangle \\ & + |01101\rangle - |01110\rangle - |10000\rangle - |10011\rangle + |10101\rangle + |10110\rangle \\ & - |11001\rangle + |11010\rangle - |11100\rangle + |11111\rangle. \quad (6.110) \end{aligned}$$

One can see that the description of the code space in terms of the so-called ‘codewords’—the logical basis states of the code space—is rather complicated. This example demonstrates how compact the description in stabilizer formalism is.

Let us evaluate explicitly the logical basis states of the code.

From the generating set of the stabilizer given above, we can construct the elements of the stabilizer explicitly.

```
In[7]:= new=Union@Join[gnr,Multiply@@@Tuples[gnr,2]];
grp=Union@Join[new,Multiply@@@Tuples[new,2]]
Length[grp]
Out[7]= {1, S1X S2X S3Y S4Y, S1X S2Y S4X S5X, S1X S2Z S3Z S4X, S1X S3X S4Z S5Z,
S1Y S2X S3Y S4Z, S1Y S2Y S3Z S5Z, S1Y S2Z S4Y, S1Y S3X S4X S5X, S1Z S2X S4Z S5Z,
S1Z S2Y S3Y S4Z, S1Z S2Z S3X S5X, S1Z S3Z S4Y S5Y, S2X S3Z S4X S5Y, S2Z S3Y S4Z S5Y}
```

Out[7]= 16

Recall that the logical Z operator is given as follows.

```
In[7]:= opZ = Multiply @@ S[jj, 3]
Out[7]= S1Z S2Z S3Z S4Z S5Z
```

We now want to get the simultaneous eigenstates of the stabilizer and the logical Z operator.

```
ops = Append[grp, opZ];
mat = Matrix@Rest@ops;
```

```
In[8]:= {val, vec} = CommonEigensystem[mat];
Style[TableForm[val[[-5 ;;]], TableAlignments -> Right], Small]
Out[8]= 1 1 -1 -1 1 1 -1 -1 -1 -1 1 1 -1 -1 1 1
1 1 1 1 -1 -1 -1 -1 -1 -1 -1 1 1 1 1 -1
1 1 1 1 -1 -1 -1 -1 -1 -1 -1 1 1 1 1 1
1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1
1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1
```

We see that the code space is spanned by the last two eigenvectors. The last one belongs to the eigenvalue 1 of the logical Z operator.

```
In[9]:= bs = Basis[S@jj];
ket0 = -vec[[-1]].bs;
ket0 // Simplify
Out[9]=  $\frac{|\underline{00000}\rangle}{4} - \frac{|\underline{00011}\rangle}{4} + \frac{|\underline{00101}\rangle}{4} - \frac{|\underline{00110}\rangle}{4} + \frac{|\underline{01001}\rangle}{4} + \frac{|\underline{01010}\rangle}{4} - \frac{|\underline{01100}\rangle}{4} - \frac{|\underline{01111}\rangle}{4}$ 
 $\frac{|\underline{10001}\rangle}{4} + \frac{|\underline{10010}\rangle}{4} + \frac{|\underline{10100}\rangle}{4} - \frac{|\underline{10111}\rangle}{4} - \frac{|\underline{11000}\rangle}{4} - \frac{|\underline{11011}\rangle}{4} - \frac{|\underline{11101}\rangle}{4} - \frac{|\underline{11110}\rangle}{4}$ 
```

```
In[10]:= ket1 = vec[[-2]].bs;
ket1 // Simplify
Out[10]=  $-\frac{1}{4} |\underline{00001}\rangle - \frac{|\underline{00010}\rangle}{4} - \frac{|\underline{00100}\rangle}{4} - \frac{|\underline{00111}\rangle}{4} - \frac{|\underline{01000}\rangle}{4} + \frac{|\underline{01011}\rangle}{4} + \frac{|\underline{01101}\rangle}{4} - \frac{|\underline{01110}\rangle}{4}$ 
 $\frac{|\underline{10000}\rangle}{4} - \frac{|\underline{10011}\rangle}{4} + \frac{|\underline{10101}\rangle}{4} + \frac{|\underline{10110}\rangle}{4} - \frac{|\underline{11001}\rangle}{4} + \frac{|\underline{11010}\rangle}{4} - \frac{|\underline{11100}\rangle}{4} + \frac{|\underline{11111}\rangle}{4}$ 
```

This confirms that they are indeed eigenstates of the logical Z operator with proper eigenvalues.

```
In[11]:= (opZ ** ket0) / ket0
          (opZ ** ket1) / ket1 // Simplify
Out[11]= 1
Out[12]= -1
```

This confirms that the logical X operator flips the logical basis states as expected.

```
In[13]:= opX ** ket0 / ket1 // Simplify
          opX ** ket1 / ket0 // Simplify
Out[13]= 1
Out[14]= 1
```

6.5 Surface Codes

In all discussions of quantum error-correction codes so far, it has been implicitly assumed that one can apply a quantum gate on any pair of qubits with uniform fidelity regardless of the spatial separation of the qubits. Clearly, it is unrealistic. In a realistic quantum computer, a qubit is only coupled directly to a few qubits in close proximity. A quantum gate on two qubits that are not directly coupled to each other is performed through virtual gate operations through other qubits between them. The fidelity of such a quantum gate is naturally much worse. *Surface codes* introduced by Kitaev in 1997 are an interesting subclass of stabilizer codes that exhibits “locality” properties ([Kitaev, 2003](#); [Dennis et al., 2002](#); [quant-ph/9811052](#); [Freedman, 2001](#); [Kitaev, 1997](#)). Surface codes rely only on local operations and measurements—one only needs to operate a quantum gate or measurement on neighboring qubits at a time. More recently, it has been shown that a fault-tolerant quantum computation with only local operations is possible based on surface codes ([Fowler et al., 2012](#)).

Another notable thing is that surface codes feature topological properties. For a surface code, operators in the stabilizer and logical operators are governed by the topology of the underlying array of qubits. It is not surprising as surface codes were derived from the models arranged on the surface of a torus ([Kitaev, 2003](#); [Dennis et al., 2002](#)) that exhibit topological order. However, the crucial point is that surface codes are tolerant to local errors ([Dennis et al., 2002](#); [Wang et al., 2003](#)). The operational locality mentioned above and the tolerance to local errors are two of the most significant advantages of surface codes.

Here we introduce two families of surface codes. They are called *toric codes* and *planar codes* after the underlying geometry of the arrangement of the qubits for the construction of the codes. Difficult to implement because of the required periodic boundary conditions, toric codes are conceptually simple and provide the key ideas of surface codes. For this reason, we first start with toric codes and then move on to planar codes, where the requirement of periodic boundary conditions is removed.

6.5.1 Toric Codes

To construct a toric code, it is convenient to suppose that the qubits are arranged on the edges of a square lattice on the surface of a torus as in Fig. 6.3. The geometry of the underlying lattice is equivalent to a square lattice on the flat plane with the opposite boundaries regarded identical. In other words, periodic boundary conditions are imposed in the horizontal and vertical directions as in Fig. 6.4. We want to construct generators of the stabilizer and logical operators.

We define two types of operators, *plaquette* and *vertex operators*. Plaquette operators are associated with the plaquettes of the underlying lattice: Consider a plaquette p , for example, denoted by an empty circle in blue in Fig. 6.4 (a). It

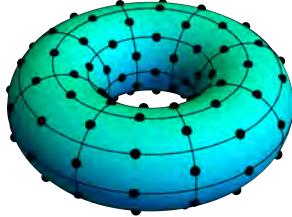


Figure 6.3: An arrangement of qubits (black dots) on a square lattice on the surface of a torus to construct a toric code.

has four edges—and hence four qubits 1, 2, 3, 4 on them—surrounding it. The plaquette operator \hat{P}_p associated with the plaquette p is defined by the product of the Pauli Z operators on the qubits on those edges,

$$\hat{P}_p = \hat{Z}_{p,1}\hat{Z}_{p,2}\hat{Z}_{p,3}\hat{Z}_{p,4}. \quad (6.111)$$

Vertex operators are attributed to the vertices of the underlying lattice: Consider a vertex v on the lattice, say, the one marked by a red open circle in Fig. 6.4 (a). It has four edges connected to the vertex and on each of the edges there resides a qubit. The vertex generator \hat{V}_v associated with the vertex v is defined by the product of the Pauli X operators on the four qubits,

$$\hat{V}_v = \hat{X}_{v,1}\hat{X}_{v,2}\hat{X}_{v,3}\hat{X}_{v,4}. \quad (6.112)$$

Obviously, all plaquette operators commute with each other, and likewise, all vertex operators commute with each other. Interesting to notice is that any plaquette operator commutes with every vertex operator because a plaquette and a vertex share either two edges or none. When they share two edges, the two Pauli Z operators from the plaquette operator commutes with the two Pauli X operators from the vertex operator.

The stabilizer of the codes is defined to be generated by the set of all plaquette and vertex operators. But how many of the generators are independent? The question is important as it determines how many qubits the code can encode. Note that a product of plaquette operators associated with neighboring plaquettes has the Pauli Z operators only on the edges of the closed boundary of the included plaquettes as illustrated by the blue closed loop in Fig. 6.4 (b). There is no non-trivial contribution from the shared edges because $\hat{Z}^2 = \hat{I}$. Similarly, a product of vertex operators associated with neighboring vertices has the Pauli X operators only on the edges crossed by the closed boundary—as illustrated by the orange dashed line in Fig. 6.4 (b)—of the involved vertices because $\hat{X}^2 = \hat{I}$. Now consider the product of all plaquette operators. The boundary embracing all the plaquettes on the lattice is the boundary of the lattice itself. However, due to the periodic

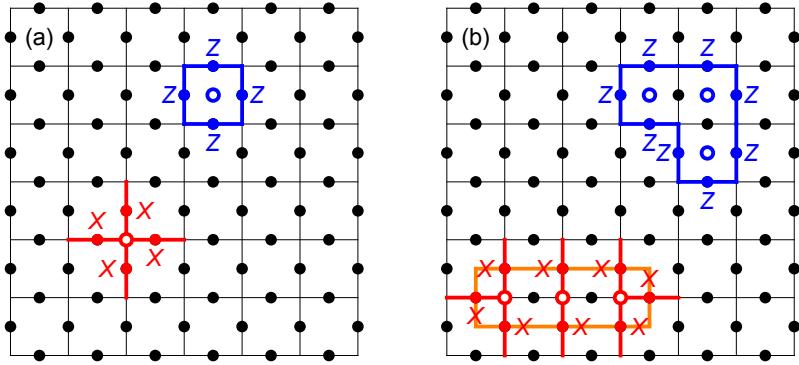


Figure 6.4: A toric code on a square lattice with periodic boundary conditions—with the opposite boundaries are regarded identical. (a) A plaquette operator (blue) and a vertex operator (red). The labels “X” and “Z” refer to the Pauli X and Z operators, respectively, on the qubit located on the edge. The plaquette and vertex operators on the lattice generate the stabilizer of the toric code. (b) Some elements of the stabilizer of a toric code. A product of neighboring plaquette operators results in a closed loop of edges (in blue) on which the Pauli Z operators act. Similarly, a product of neighboring vertex operators results in a closed loop of dual edges (dashed in orange) on which the Pauli X operators act.

boundary conditions, the boundary is trivial. That is, the product of all plaquette operators is the identity, $\prod_p \hat{P}_p = \hat{I}$. In the same manner, it follows that the product of all vertex operators is the identity as well, $\prod_v \hat{V}_v = \hat{I}$. Therefore, one plaquette or vertex operator can be expressed as the product of the rest of the same type. In other words, all but one plaquette operators are independent of each other. It is also the case for vertex operators.

On an $L \times L$ lattice, there are $2L^2$ edges. On the other hand, there are L^2 plaquettes and L^2 vertices, and overall there are $2L^2$ plaquette or vertex operators. However, as pointed out above, among L^2 plaquette operators, only $L^2 - 1$ of them are independent. Likewise, out of L^2 vertex operators, only $L^2 - 1$ of them are independent of each other. In short, the stabilizer of the code is generated by $2(L^2 - 1)$ independent generators. It implies that the code space is 2^2 -dimensional—the code can encode two logical qubits.

As the code encodes two logical qubits, we need to construct two logical Z operators and two logical X operators. Recall (see Section 6.3.2) that logical operators must commute with all elements of the stabilizer—they must preserve the code space—but do not belong to the stabilizer. Let us consider a line of edges running all the way from the bottom boundary to the top boundary as depicted by a thick vertical in blue in Fig. 6.5 (a). Let \bar{Z}_1 be the product of the Pauli Z operators on all edges along the line. It follows that \bar{Z}_1 commute with all elements of the stabilizer because the line shares two edges (if at all) with any

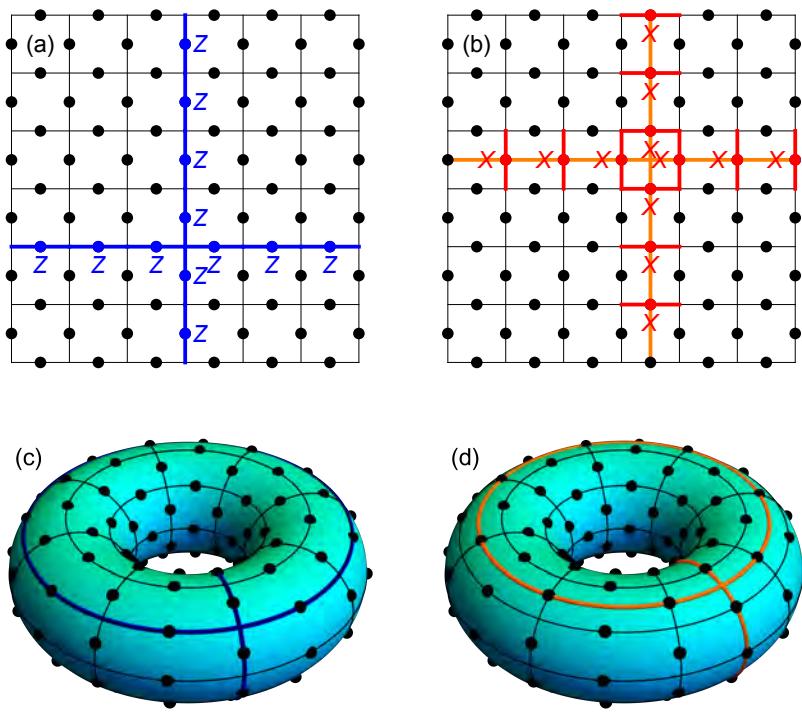


Figure 6.5: Logical operators in a toric code. (a) Two inequivalent logical Z operators that correspond to two homologically different loops encircling the hole of the torus (c). (b) Two inequivalent logical X operators corresponding to the two homologically different loops on the surface of torus (d).

vertex operator—it commutes trivially with all plaquette operators. However, it cannot be a member of the stabilizer—it cannot be expressed as a product of plaquette operators—because the line cannot be a boundary of a region on the lattice. This can be seen more clearly on the surface of the underlying torus shown in Fig. 6.5 (c). The line is a closed loop through the hole of the torus, and it is homologically different from the boundary loop of any region of the surface. Therefore, we can take \bar{Z}_1 as one of the logical Z operators. The other logical Z operator can be constructed from a line running from the left to right boundary as depicted by the thick horizontal line in blue in Fig. 6.5 (a). That is, we define \bar{Z}_2 to be the product of the Pauli Z operators on all edges along the line. The line encloses the hole of the torus as illustrated in Fig. 6.5 (c). Following the same arguments as above, we see that \bar{Z}_2 commute with all elements of the stabilizer but does not belong to the stabilizer. We take \bar{Z}_2 as the second logical Z operator.

Logical X operators can also be constructed in a similar way. Consider a line running vertically from the bottom to top boundary through the plaquettes as depicted by the vertical orange dashed line in Fig. 6.5 (b). We define \bar{X}_1 to be the product of the Pauli X operators on all the edges crossing the line. \bar{X}_2 is defined in the same manner with a line through plaquettes from the left to right boundary of the lattice. For the same reason given above, the two operators \bar{X}_1 and \bar{X}_2 commute with all elements of the stabilizer but are not members of the stabilizer. Therefore, we take them as logical X operators of the code.

Recall that the choice of logical operators is not unique. In this case, difference choice of logical operators correspond to different lines connecting opposite boundaries. The lines do not need to be straight as long as they connect opposite boundaries. For example, instead of the straight line for \bar{Z}_1 , one can choose any line connecting the bottom and top boundaries. The resulting operator \bar{Z}'_1 is obtained from \bar{Z}_1 by multiplying plaquette operators. In this sense, all logical operators associated with lines that can be smoothly deformed to each other on the surface of the underlying torus can be regarded as equivalent.

Consider a toric code on a 3x3 square lattice on the surface of a torus.

```
$L = 3;
jj = Range[0, $L - 1];
```

The qubits on the horizontal and vertical edges are labelled by S and T, respectively.

```
Let[Qubit, S, T]
```

This defines plaquette operators.

```
A[{i_, j_}] :=
  S[i, j, 3] ** S[i, Mod[j + 1, $L], 3] ** T[i, j, 3] ** T[Mod[i + 1, $L], j, 3]
```

Here are the *independent* plaquette operators.

```
In[7]:= AA = Most@Flatten@Table[A@{i, j}, {i, jj}, {j, jj}]
Out[7]= {S0,0z, S0,1z, T0,0z, T1,0z, S0,1z, S0,2z, T0,1z, T1,1z, S0,0z, S0,2z, T0,2z, T1,2z, S1,0z, S1,1z, T1,0z, T2,0z, S1,1z, S1,2z, T1,1z, T2,1z, S1,0z, S1,2z, T1,2z, T2,2z, S2,0z, S2,1z, T0,0z, T2,0z, S2,1z, S2,2z, T0,1z, T2,1z}
```

```
In[8]:= Length[AA]
```

```
Out[8]= 8
```

This defines vertex operators.

```
B[{i_, j_}] := 
  S[i, j, 1] ** S[Mod[i - 1, $L], j, 1] ** T[i, j, 1] ** T[i, Mod[j - 1, $L], 1]
```

Here are the *independent* vertex operators.

```
In[9]:= BB = Most@Flatten@Table[B@{i, j}, {i, jj}, {j, jj}]
Out[9]= {S0,0x, S0,1x, T0,0x, T0,2x, S0,1x, S2,1x, T0,0x, T0,1x, S0,2x, S2,2x, T0,1x, T0,2x, S0,0x, S1,0x, T1,0x, T1,2x, S0,1x, S1,1x, T1,0x, T1,1x, S0,2x, S1,2x, T1,1x, T1,2x, S1,0x, S2,0x, T2,0x, T2,2x, S1,1x, S2,1x, T2,0x, T2,1x}
```

```
In[10]:= Length[BB]
```

```
Out[10]= 8
```

The plaquette operators commute with vertex operators.

```
In[11]:= Outer[GottesmanTest, AA, BB] // MatrixForm
Out[11]/MatrixForm=
(1 1 1 1 1 1 1 1 1
 1 1 1 1 1 1 1 1
 1 1 1 1 1 1 1 1
 1 1 1 1 1 1 1 1
 1 1 1 1 1 1 1 1
 1 1 1 1 1 1 1 1
 1 1 1 1 1 1 1 1
 1 1 1 1 1 1 1 1)
```

Here are two logical Z operators of the code.

```
In[12]:= opZ = {
  Multiply @@ Table[S[i, 1, 3], {i, jj}],
  Multiply @@ Table[T[0, j, 3], {j, jj}]
}
Out[12]= {S0,1z, S1,1z, S2,1z, T0,0z, T0,1z, T0,2z}
```

```
In[13]:= Outer[GottesmanTest, opZ, AA] // MatrixForm
Outer[GottesmanTest, opZ, BB] // MatrixForm
Out[13]/MatrixForm=
(1 1 1 1 1 1 1 1
 1 1 1 1 1 1 1 1)
```

```
Out[14]/MatrixForm=
(1 1 1 1 1 1 1 1
 1 1 1 1 1 1 1 1)
```

Here are two logical X operators of the code.

```
In[15]:= opX = {
  Multiply @@ Table[T[i, 2, 1], {i, jj}],
  Multiply @@ Table[S[1, j, 1], {j, jj}]
}
Out[15]= {T0,2x, T1,2x, T2,2x, S1,0x, S1,1x, S1,2x}
```

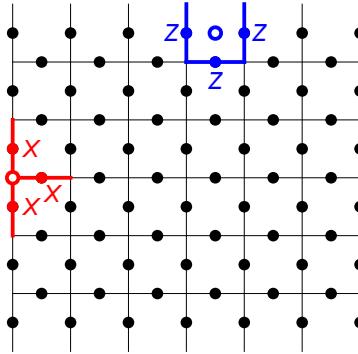


Figure 6.6: (A planar surface code on a square lattice with rough and smooth boundaries. Shown are a plaquette operator associated with a plaquette (blue empty circle) at the rough boundary at the top and a vertex operator associated with a vertex (red empty circle) at the smooth boundary on the left. The labels “ X ” and “ Z ” refer to the Pauli X and Z operators, respectively, on the qubit located on the edge.

```
In[=]:= Outer[GottesmanTest, opX, AA] // MatrixForm
Outer[GottesmanTest, opX, BB] // MatrixForm
Out[=]//MatrixForm=

$$\begin{pmatrix} 1 & 1 & 1 & 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 & 1 & 1 & 1 \end{pmatrix}$$

Out[=]//MatrixForm=

$$\begin{pmatrix} 1 & 1 & 1 & 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 & 1 & 1 & 1 \end{pmatrix}$$

```

]]

6.5.2 Planar Codes

Now let us turn to planar codes. For a planar code, we consider a square lattice without imposing periodic boundary condition. Instead, we need two different types of boundaries. In Fig. 6.6, the left and right boundaries are ending with edges while the top and bottom boundaries are terminated with vertices. The former are called *smooth boundaries* and the latter *rough boundaries*.

The plaquette and vertex operators associated with plaquettes and vertices interior of the lattice are defined the same as for toric codes. However, plaquettes and vertices at the boundaries have different surrounding edges, the corresponding operators require slightly modified definitions. A vertex on a rough edge has only three edges around it, and the plaquette operator associated with such a vertex is given by a product of three Pauli Z operators,

$$\hat{P}_p = \hat{Z}_{p,1}\hat{Z}_{p,2}\hat{Z}_{p,3}. \quad (6.113)$$

Likewise, a vertex at smooth boundaries has only three edges connected to it, and the vertex operator associated with such a vertex is given by a product of three Pauli X operators

$$\hat{V}_v = \hat{X}_{v,1}\hat{X}_{v,2}\hat{X}_{v,3}. \quad (6.114)$$

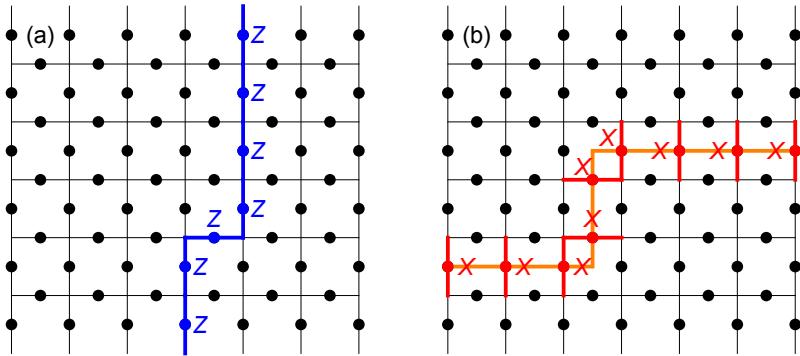


Figure 6.7: A logical Z operator (a) and logical X operator (b) in a planar surface code.

All plaquette and vertex operators in the interior and at the boundary of the lattice generates the stabilizer. Without the periodic boundary condition imposed in toric codes, all plaquette and vertex operators are independent of each other. How many of them are there? For a $L \times L$ square lattice with two rough and two smooth boundaries, there are $L(L - 1)$ plaquettes and $L(L - 1)$ vertices. On the other hand, there are $L^2 + (L - 1)^2$ edges and hence the same number of physical qubits on the lattice. It implies that the planar code can encode one logical qubit.

Let us construct logical operators \bar{Z} and \bar{X} on the code space. Consider a line running from the bottom to top boundaries. An example is shown in Fig. 6.7 (a). Recall that both are rough boundaries. The line share two edges, if at all, with a vertex operator regardless whether it is in the interior or at the smooth boundaries. Therefore, if we define \bar{Z} to be the product of the Pauli Z operators on the edges along the line, then \bar{Z} commutes with all the generators of the stabilizer. Ending at opposite boundaries, however, \bar{Z} cannot be generated by any combination of plaquette operators. This concludes that \bar{Z} can be chosen as the logical Z operator. Similarly, we can choose a line running through plaquettes from the left to right edges as depicted by a orange dashed line in Fig. 6.7 (b). We define the logical operator \bar{Z} to be the product of the Pauli X operators on the edges crossing the line. As it shares two edges, if at all, with any plaquette (either in the interior or at the rough boundaries), \bar{X} certainly commutes with all plaquette operators, but it is independent of any plaquette or vertex operators.

So far we have constructed surface codes associated with square lattices, but it is not restricted to square lattices (Dennis *et al.*, 2002). One can construct a surface code for any tessellation of a surface. One can also consider surfaces of higher genus. For a closed orientable surface of genus g , $2g$ qubits can be encoded. For planar codes, we may consider a surface with e distinct rough edges separated by e distinct smooth edges. Then $e - 1$ qubits can be encoded, associated with the lines that connect one rough edge with any of the others. Punching holes in the lattice is another way to encode more logical qubits as shown in Fig. 6.8. Suppose

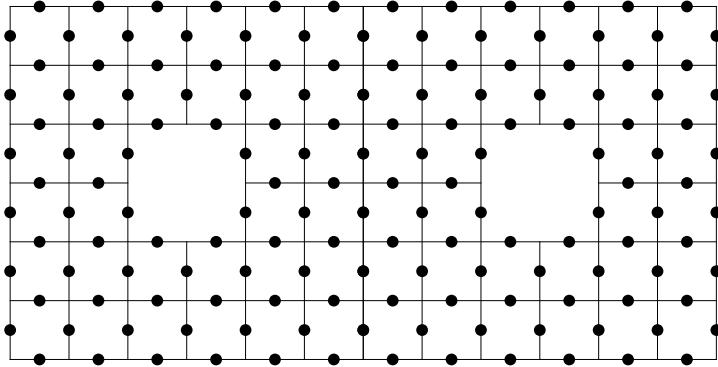


Figure 6.8: A surface code on a lattice with a smooth outer boundary and two holes with smooth boundaries. It can encode two qubits.

that the outer boundary of the lattice is smooth and there are h holes with smooth boundaries. Then the code can encode h qubits. For each hole, a closed loop on the lattice that encloses the hole is associated with a logical Z operator, and a line through plaquettes—a line on the dual lattice—from the boundary of the hole to the outer boundary is associated with a logical X operator.

6.5.3 Recovery Procedure

As a special type of stabilizer code, the recovery procedure of surface codes also follows the general prescription: first measure the generators to detect error syndrome and apply necessary Pauli operators to correct the affected qubits.

For surface codes, the generators of the stabilizer are either plaquette or vertex operators. The local measurement of those operators can be achieved by using ancilla qubits on plaquettes or vertices of the underlying lattice (Fowler *et al.*, 2012). Let us first consider a particular plaquette. We assume an ancillary qubit at the center of the plaquette, and prepare it initially in the state $|0\rangle$. We then apply a unitary gate described by the following quantum circuit model

$$\begin{array}{c}
 |x_1\rangle \xrightarrow{\quad} |x_1\rangle \\
 |x_2\rangle \xrightarrow{\quad} |x_2\rangle \\
 |x_3\rangle \xrightarrow{\quad} |x_3\rangle \\
 |x_4\rangle \xrightarrow{\quad} |x_4\rangle \\
 |0\rangle \xrightarrow{\bigoplus} |x_1 \oplus x_2 \oplus x_3 \oplus x_4\rangle
 \end{array} . \quad (6.115)$$

The output state of the ancilla qubit is given by

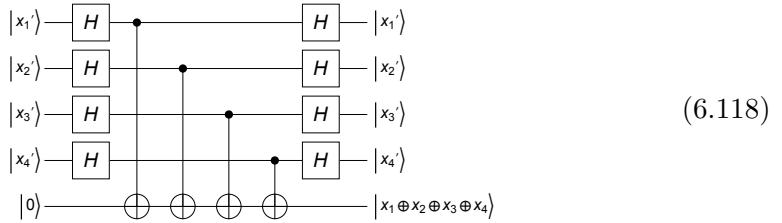
$$|x_1 \oplus x_2 \oplus x_3 \oplus x_4\rangle \quad (6.116)$$

if the physical qubits 1, 2, 3, 4 on the surrounding edges is initially in the state $|x_1 x_2 x_3 x_4\rangle$. Therefore, measuring the ancilla qubit in the standard basis gives one of the eigenvalues

$$x_1 \oplus x_2 \oplus x_3 \oplus x_4 = \pm 1 \quad (6.117)$$

of the plaquette operator $\hat{P} = \hat{Z}_1 \hat{Z}_2 \hat{Z}_3 \hat{Z}_4$.

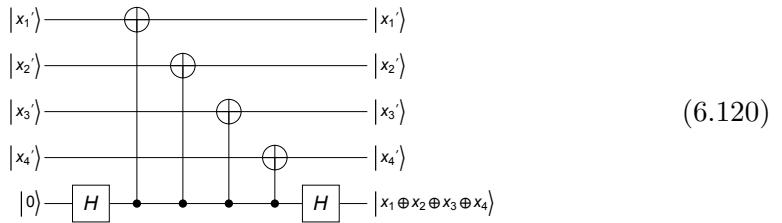
The measurement of the vertex operator $\hat{V} = \hat{X}_1 \hat{X}_2 \hat{X}_3 \hat{X}_4$ associated with a given vertex with four edges 1, 2, 3, 4 connected to it can be achieved through a slightly different unitary gate. We assume an ancilla qubit located at the vertex. Recalling that that $\hat{V} = \hat{H}^{\otimes 4} \hat{P} \hat{H}^{\otimes 4}$, one can see that the unitary gate described by the quantum circuit model of the form



followed by a measurement on the ancilla qubit leads to a measurement of the vertex operator \hat{V} . Here the states $|x'\rangle$ for $x = 0, 1$ denote the eigenstates

$$|0'\rangle \equiv |+\rangle := \frac{|0\rangle + |1\rangle}{\sqrt{2}}, \quad |1'\rangle \equiv |- \rangle := \frac{|0\rangle - |1\rangle}{\sqrt{2}} \quad (6.119)$$

of the Pauli X operator belonging to the eigenvalue ± 1 . They correspond to $|x\rangle$ but are defined in the rotated frame, $|x'\rangle = \hat{H}|x\rangle$. Using the identities in Eq. (2.43), one can simplify the above quantum circuit model to the form (see Problem 6.9)



Plaquette and vertex operators at the boundaries can be achieved in the same way as well because the scheme works for any number of physical qubits.

Error syndrome of surface code features interesting topological nature. A phase-flip error \hat{Z} on a qubit, say, qubit 1 in Fig. 6.9 (a) can be diagnosed by measuring two vertex operators at the two ends of the edge where the corrupted qubit located on. The error is corrected simplify by applying \hat{Z} on qubit 1. Note that a series of phase-flip errors on qubits 2, 3, and 4 give the same error syndrome, of course, diagonalized by the same vertex operators as the phase-flip error on qubit 1. As we

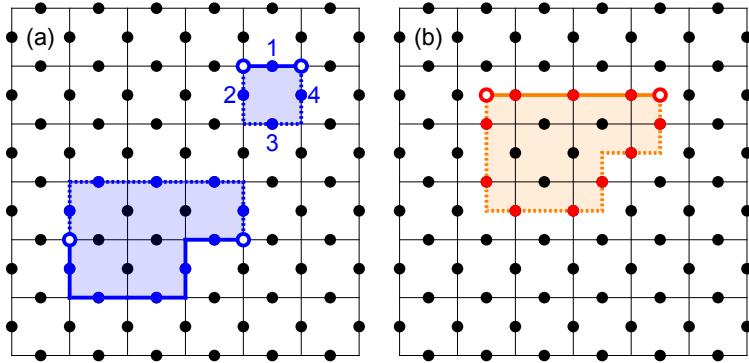


Figure 6.9: (a) Phase-flip errors (blue dots on thick blue edges) and vertex defects (blue empty circles) associated with them. (b) Bit-flip errors (red dots on thick orange lines) and vertex defects (red empty circles) associated with them.

discussed the general error recovery procedure for stabilizer codes in Section 6.4, even if the actual errors occurred on qubits 2, 3, and 4, the errors can still be fixed by applying \hat{Z} on qubit 1. In this case, it is because the application of \hat{Z} on qubit 1, 2, 3, and 4 corresponds to applying the plaquette operator $\hat{P} = \hat{Z}_1\hat{Z}_2\hat{Z}_3\hat{Z}_4$ associated with the plaquette (shaded in blue in Fig. 6.9 (a)) surrounded by the four edges. Recall that plaquette operators belong to the stabilizer of the surface code (in this particular example, we assume a toric code), and they operate trivially on the code space.

Another more general example is shown in the left lower corner of Fig. 6.9 (a). It depicts two cases of correlated phase-flip errors, one on the qubits on the blue solid line and the other on the qubits on the blue dashed line. Both cases give the same error syndrome, detected by the vertex operators at the two ends shared by the two lines. In fact, any line connecting the same ends give the same error syndrome as long as the line can be smoothly deformed on the surface of torus—recall that the square lattice with periodic boundary conditions on flat plane is equivalent to the one the surface of a torus.^{6.9} In this sense, it is convenient to regard that phase-flip errors on neighboring edges create *vertex defects* at the ends of the line of the affected edges. Vertex defects can be detected by measuring vertex operators. The phase-flip errors associated with a line connecting a pair of vertex defects^{6.10} can be corrected by applying \hat{Z} on the qubits on any line that can be smoothly deformed from the line.

Similarly, bit-flip errors \hat{X} on a line segment through plaquettes can be regarded to create *plaquette defects* at the end of the line segment. An example is illustrated in Fig. 6.9 (b). The bit-flip errors on qubits along the orange solid line and those along the orange dashed line give the same error syndrome because the two lines

^{6.9}In planar codes, a line of phase-flip errors should not hit a rough boundary.

^{6.10}In toric codes, vertex defects can only be created in pairs. In planar codes, single vertex defects can be created when the line of phase-flip errors ends at a rough boundary.

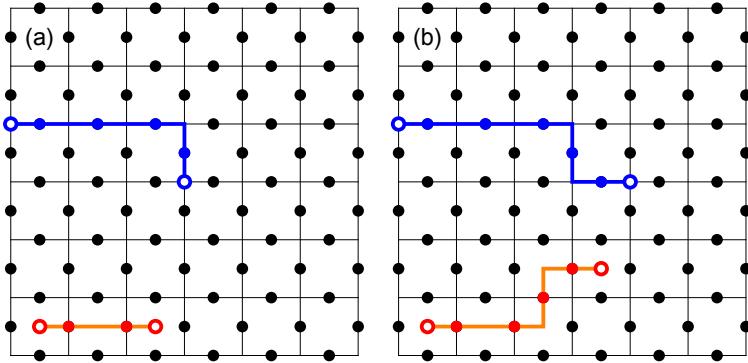


Figure 6.10: Vertex (blue empty circles) and plaquette (red empty circles) defects before (a) and after (b) defects have moved. One can move a vertex defect by applying \hat{Z} on a qubit next to it; and a plaquette defect by applying \hat{X} next to it.

share the same ends and can be smoothly deformed to each other without cutting the lines.

Error syndrome in surface codes has even deeper topological aspects, and further, the vertex and plaquette defects have physical interpretation as “particles” (Kitaev, 2003). Consider examples illustrated in Fig. 6.10. By applying \hat{Z} on a qubit next to one of the vertex defects in Fig. 6.10 (a), we can “move” the vertex defect to a neighboring vertex as in Fig. 6.10 (b). Similarly, one can also move a plaquette defect by applying \hat{X} on a qubit next to the plaquette defect. At the bottom of Fig. 6.10 (a), there are two plaquette defects associated with a line of bit-flip errors (orange line). The plaquette defect on the right has moved up by one lattice unit and to the right by one lattice unit as in Fig. 6.10 (b). It has been achieved by applying two \hat{X} operators consecutively on two qubits next to the plaquette defect.

The motion of vertex and plaquette defects (or particles) features fascinating topological properties as illustrated in Fig. 6.11. Consider first the lower example in Fig. 6.11. The vertex defect at the right end of a line of phase-flip errors is moved around two plaquettes. The overall effect is equivalent to applying two vertex operators enclosed by the path and trivial on the code space. Now let us turn to the upper example in Fig. 6.11 (b). In this case, the vertex defect on the right travels around four plaquettes and returns to its original vertex. However, a crucial difference here is that the path encloses a plaquette defect. What is the overall effect of the motion of the vertex defect? One can clearly see in this particular example that the path of the vertex defect shares one edge (at the point where the orange and blue line cross) with the line (in orange) of bit-flip errors. However, this is completely general and the path must share an odd number of edges with the line of bit-flip errors as long as it closes an odd number of plaquette defects. Then, due to the anti-commutation of \hat{X} and \hat{Z} , such a path causes a phase shift of -1 . Topologically, it is said that the path cannot shrink smoothly

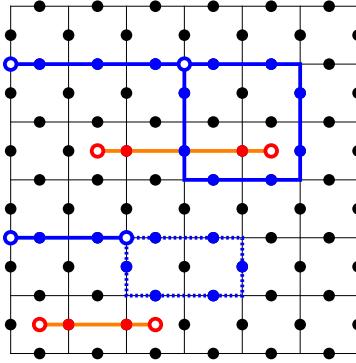


Figure 6.11: Exchange of vertex and plaquette defects. In the lower example, a vertex defect moves and returns to its original vertex without enclosing any other defect. It has no effect. In the upper example, the path along which the right vertex defect has traveled encloses a plaquette vertex. It causes a phase shift of -1 .

to a point because of the enclosed plaquette defects. In many other respects, vertex and plaquette defects behave like particles of different species with exotic statistical properties.

The manipulation of vertex and plaquette defects described above is an example of topological quantum computation, the account of which has long been deferred from Section 3.3. As a matter of fact, it was the first proposal for topological quantum computation, and has inspired a huge number of works on the subject. More recently, the idea was further developed for topological quantum computation with Majorana fermions—fermions that are anti-particles of their own—on more realistic physical systems (Kitaev, 2001; Alicea *et al.*, 2011). The latter works lead to a fairly dramatic quest for Majorana fermions. The existence of Majorana fermions as elementary excitations in condensed matter systems have accumulated a growing number of experimental evidences (Mourik *et al.*, 2012; Deng *et al.*, 2012; Das *et al.*, 2012; Nadj-Perge *et al.*, 2014).

Problems

- 6.1. Let \mathcal{V} be a code space. Let \mathcal{S} is the stabilizer subgroup of the Pauli group with respect to \mathcal{V} . Show that any two elements of \mathcal{S} commute.
- 6.2. Let $\hat{G}_1, \hat{G}_2, \dots, \hat{G}_k$ be elements of the Pauli group $\mathcal{P}(n)$ on n qubits. Show that $\hat{G}_1, \hat{G}_2, \dots, \hat{G}_k$ are *independent* if and only if $|\hat{G}_1\rangle, |\hat{G}_2\rangle, \dots, |\hat{G}_k\rangle$ are *linearly independent* of each other in $(\mathbb{Z}_2)^{2n}$ regarded as a vector space over \mathbb{Z}_2 . $|\hat{G}_j\rangle$ are defined by the correspondence (6.41).

6.3. Let $\hat{G}_1, \hat{G}_2, \dots, \hat{G}_k$ be mutually *independent* elements of the Pauli group $\mathcal{P}(n)$ on n qubits. Show that there exists an element \hat{G} in $\mathcal{P}(n)$ such that \hat{G} anti-commutes with one and only one of $\hat{G}_1, \hat{G}_2, \dots, \hat{G}_k$ and commutes with the rest.

6.4. Identify the elements of the single-qubit Clifford group $\mathcal{C}(1)$.

Hint: In total, there are 192 elements in $\mathcal{C}(1)$, generated by the Hadamard gate \hat{H} and the quadrant phase gate \hat{Q} . Note also that the elements appear with physically irrelevant phase factors $\pm 1, \pm i, \pm e^{i\pi/4}, \pm e^{-i\pi/4}$. This example illustrates how efficient the description of a group in terms of its generators is.

6.5. Let \hat{A} and \hat{B} be the CNOT gates on the same pair of qubits but with the control and target qubit exchanged. Show that \hat{A} and \hat{B} are not independent generators of the Clifford group.

6.6. Let \hat{U} be an element of the $(n+1)$ -qubit Clifford group $\mathcal{C}(n+1)$. Show that by applying single-qubit operations and/or rearranging the qubits if necessary, one can always put both $\hat{U}(\hat{Z} \otimes \hat{J})\hat{U}^\dagger$ and $\hat{U}(\hat{X} \otimes \hat{J})\hat{U}^\dagger$ simultaneously in the form

$$\hat{U}(\hat{Z} \otimes \hat{J})\hat{U}^\dagger = \hat{X} \otimes \hat{A}, \quad \hat{U}(\hat{X} \otimes \hat{J})\hat{U}^\dagger = \hat{Z} \otimes \hat{B}, \quad (6.121)$$

where $\hat{J} := \hat{I}^{\otimes n}$ and $\hat{A}, \hat{B} \in \mathcal{P}(n)$.

Note: It is also possible to put them simultaneously in the form

$$\hat{U}(\hat{Z} \otimes \hat{J})\hat{U}^\dagger = \hat{X} \otimes \hat{A}, \quad \hat{U}(\hat{X} \otimes \hat{J})\hat{U}^\dagger = \hat{I} \otimes \hat{B}. \quad (6.122)$$

6.7. Let \hat{U} be an element of the $(n+1)$ -qubit Clifford group $\mathcal{C}(n+1)$. Define an operator \hat{V} acting on n qubits by

$$\hat{V}|y\rangle := \sum_{x=0}^{2^n-1} |x\rangle \langle \langle 0| \otimes \langle x|) \hat{U}(|0\rangle \otimes |y\rangle). \quad (6.123)$$

Show that \hat{V} is an element of n -qubit Clifford group $\mathcal{C}(n)$.

6.8. Consider a toric code associated with a 3×3 square lattice on the surface of a torus. Find the four logical basis states of the code space.

Remark: There is no need to list the logical basis states explicitly to implement a surface code. This is a heuristic exercise.

6.9. Show that the two quantum circuit models in Eqs. (6.118) and (6.120) are equivalent.

Hint: Recall the identities in Eq. (2.43).

Appendix A

Linear Algebra

• August 9, 2021 (v1.19)

Linear algebra is an elementary language to describe quantum mechanics mathematically. This appendix summarizes the concepts, definitions, theorems, and properties of linear algebra that are frequently used in quantum information physics. In most cases, we omit rigorous proofs to focus on main concepts.

Many textbooks on linear algebra focus on arithmetic techniques related to properties of matrices, such as the Gauss elimination and the formula of determinant. In most areas of physics, notably quantum mechanics, more relevant are the fundamental concepts and algebraic structures of vector spaces and linear operators. [Lang \(1987\)](#)—an abridged edition [Lang \(1986\)](#) is also available—is one of the textbooks that introduce and discuss the latter subjects at a level adequate to physicists.

A.1 Vectors

A.1.1 Vector Space

One of the most distinguished features of quantum states compared with classical states is superposition inherited from the wave-particle duality. It is thus natural to describe quantum states mathematically by vectors. Vectors can be multiplied by numbers (called scalars) and added with each other, exactly the way the superposition principle dictates. We first need a field, a set of scalars with addition and multiplication.

Definition A.1 (field) A set \mathbb{F} of elements is called a *field* if it satisfies the following conditions:

- (a) (addition) If $x, y \in \mathbb{F}$, then $x + y \in \mathbb{F}$.
- (b) (multiplication) If $x, y \in \mathbb{F}$, then $xy \in \mathbb{F}$.

- (c) (zero) $0 \in \mathbb{F}$ and $1 \in \mathbb{F}$.
- (d) (inverse) If $x \in \mathbb{F}$, then $-x \in \mathbb{F}$.
- (e) (inverse) If $x \in \mathbb{F}$ and $x \neq 0$, then $x^{-1} \in \mathbb{F}$.

The elements of the given field are called the *scalars*.

In the simplest terms, vectors represent physical quantities with both magnitude and direction. In quantum mechanics, more important feature of vectors is superposition. Here is the formal definition with mathematical rigor.

Definition A.2 (*vector space*) A set \mathcal{V} of elements is called a *vector space* over a field \mathbb{F} , if it satisfies the following conditions:

- (a) If $v_1, v_2 \in \mathcal{V}$, then $v_1 + v_2 \in \mathcal{V}$.
- (b) If $v_1 \in \mathcal{V}$ and $x \in \mathbb{F}$, then $xv_2 \in \mathcal{V}$.
- (c) If $v_1, v_2, v_3 \in \mathcal{V}$, then $(v_1 + v_2) + v_3 = v_1 + (v_2 + v_3)$.
- (d) There is an element $0 \in \mathcal{V}$ (a *null vector*) such that for all $v_2 \in \mathcal{V}$ $0 + v_2 = v_2 + 0 = v_2$. Note that both “zero” in \mathbb{F} or the null vector in \mathcal{V} are denoted by ‘0’.
- (e) For a given $v_2 \in \mathcal{V}$, there exists $-v_2 \in \mathcal{V}$ such that $v_2 + (-v_2) = 0$. The subtraction between vectors are defined by $v_1 - v_2 := v_1 + (-v_2)$.
- (f) For $x, y \in \mathbb{F}$ and $v_1, v_2 \in \mathcal{V}$,

$$x(v_1 + v_2) = xv_1 + xv_2, \quad (\text{A.1a})$$

$$(x + y)v_2 = xv_2 + yv_2, \quad (\text{A.1b})$$

$$(xy)v_2 = x(yv_2). \quad (\text{A.1c})$$

The elements of a vector space are called *vectors*.

Common examples include vector spaces over the fields of rational numbers (\mathbb{Q}), real numbers (\mathbb{R}), complex numbers (\mathbb{C}), and quaternions (\mathbb{H}). Note that the set of integer numbers (\mathbb{Z}) with the standard arithmetic rules is not a field. As is the case mostly in quantum mechanics, *vector spaces will be assumed to be over the field of complex numbers \mathbb{C} throughout this book unless mentioned otherwise*.

A.1.2 Hermitian Product

In the definition of vector space, the multiplication of vectors has not be defined. The *inner product* gives a special kind of multiplication between vectors and provides the vector space with a geometric structure, i.e., the orthogonality of vectors.

Inner product is usually a bilinear product. In many fields of physics (e.g., quantum mechanics), however, we will be dealing with vector spaces over \mathbb{C} (the field of complex numbers). To preserve the notion of positive definiteness, we need to adopt a slightly different definition of inner product. This modified inner product is called a “Hermitian product” to distinguish it from a usual inner product.

Definition A.3 (Hermitian product) Let \mathcal{V} be a vector space. A *Hermitian product* on \mathcal{V} is a function $\langle \cdot, \cdot \rangle$ from $\mathcal{V} \times \mathcal{V}$ to \mathbb{C} satisfying the following conditions:

- (a) For all $u, v, w \in \mathcal{V}$, $\langle u, v + w \rangle = \langle u, v \rangle + \langle u, w \rangle$.
- (b) For all $z \in \mathbb{C}$ and $v, w \in \mathcal{V}$, $\langle zv, w \rangle = z^* \langle v, w \rangle$ and $\langle v, zw \rangle = z \langle v, w \rangle$.
- (c) For all $v, w \in \mathcal{V}$, $\langle v, w \rangle = \langle w, v \rangle^*$.
- (d) $\langle v, v \rangle \geq 0$ for all $v \in \mathcal{V}$, and $\langle v, v \rangle > 0$ if $v \neq 0$.^{A.1}

The geometric structure due to the Hermitian product allows to define the magnitude of vectors which is important in the probabilistic interpretation of quantum mechanics (see Section 1.3). It also enables to quantify how close two state vectors are through the notion of *fidelity*. The fidelity between two vectors v and w is defined to be $|\langle v, w \rangle|$.

A.1.3 Basis

Definition A.4 (linear independence) Let \mathcal{V} be a vector space. Vectors v_1, \dots, v_n in \mathcal{V} are said to be *linearly dependent* of each other if there exists a solution $z_1, \dots, z_n \in \mathbb{C}$ to the equation

$$z_1 v_1 + \dots + z_n v_n = 0. \quad (\text{A.2})$$

If not, they are said to be *linearly independent*.

Definition A.5 (basis) Let \mathcal{V} be a vector space. If every element of \mathcal{V} is a linear combination of v_1, \dots, v_n , then v_1, \dots, v_n are said to *span* (or *generate*) the vector space \mathcal{V} . The set $\{v_1, \dots, v_n\} \subset \mathcal{V}$ is called a *basis* of \mathcal{V} if v_1, \dots, v_n span \mathcal{V} and are linearly independent. The number of elements in a basis of \mathcal{V} is called the *dimension* of \mathcal{V} and denoted by $\dim \mathcal{V}$.

Quantum states are described by a state vector in a Hilbert space. Hilbert space is a vector space, usually infinite dimensional, with additional analytic properties provided by the notion of completeness. However, as long as the dimension is finite, there is no distinction between Hilbert space and vector space. Unless mentioned otherwise explicitly, we assume that vector spaces are finite dimensional.

^{A.1}This positive definiteness is included in the definition as it is required in most applications in quantum mechanics.

When a basis (recall Definition A.5) $\{v_1, v_2, \dots, v_n\}$ spanning \mathcal{V} satisfies

$$\langle v_i, v_j \rangle = \delta_{ij}, \quad (\text{A.3})$$

it is called an *orthonormal basis*. For a finite dimensional vector space \mathcal{V} , one can always find an orthogonal basis as long as $\mathcal{V} \neq \{0\}$.

A choice of basis is arbitrary and one can change the basis to another: Suppose that $\mathcal{A} = \{v_1, v_2, \dots, v_n\}$ and $\mathcal{B} = \{w_1, w_2, \dots, w_n\}$ are two bases of the same vector space \mathcal{V} . As both \mathcal{A} and \mathcal{B} are bases, each vector $w_j \in \mathcal{B}$ is expanded in $v_i \in \mathcal{A}$ (and vice versa):

$$w_j = \sum_i v_i U_{ij}, \quad U_{ij} \in \mathbb{C}. \quad (\text{A.4})$$

The matrix $U := [U_{ij}]$ composed of these coefficients characterizes the relation between the two bases, and must be invertible since the elements in each basis are linearly independent of each other. The relation is particularly simple when the bases are *orthonormal*: As \mathcal{A} is orthonormal, the coefficients U_{ij} can be obtained by

$$U_{ij} = \langle v_i, w_j \rangle. \quad (\text{A.5})$$

More importantly, one can show that U is a *unitary* matrix—see also Theorem A.15.

A.1.4 Representations

Given a fixed basis $\{v_1, v_2, \dots, v_n\}$ of a vector space \mathcal{V} , any vector $\alpha \in \mathcal{V}$ is *uniquely* specified by the coefficients $\alpha_j \in \mathbb{C}$ in the expansion

$$\alpha = v_1 \alpha_1 + v_2 \alpha_2 + \dots + v_n \alpha_n. \quad (\text{A.6})$$

The column vector consisting of $\alpha_1, \dots, \alpha_n$ is said to be the *representation* of the vector α in the basis, and denoted by

$$\alpha \doteq \begin{bmatrix} \alpha_1 \\ \alpha_2 \\ \vdots \\ \alpha_n \end{bmatrix}. \quad (\text{A.7})$$

When the basis $\{v_1, v_2, \dots, v_n\}$ is *orthonormal*, the expansion coefficients α_j are obtained directly by means of the Hermitian product, $\alpha_j = \langle v_j, \alpha \rangle$. Hence, the vector is represented by the expansion

$$\alpha = \sum_j v_j \langle v_j, \alpha \rangle \quad (\text{A.8})$$

or, equivalently, by the column vector

$$\alpha \doteq \begin{bmatrix} \langle v_1, \alpha \rangle \\ \langle v_2, \alpha \rangle \\ \vdots \\ \langle v_n, \alpha \rangle \end{bmatrix}. \quad (\text{A.9})$$

Consider another vector $\beta \in \mathcal{V}$ and suppose that $\beta_j := \langle v_j, \beta \rangle$ is its representation in the same orthonormal basis. Then, the Hermitian product $\langle \alpha, \beta \rangle$ can be evaluated using their column-vector representations

$$\langle \alpha, \beta \rangle = \sum_j \alpha_j^* \beta_j = [\alpha_1^* \quad \alpha_2^* \quad \cdots \quad \alpha_n^*] \begin{bmatrix} \beta_1 \\ \beta_2 \\ \vdots \\ \beta_n \end{bmatrix}, \quad (\text{A.10})$$

where we have used the identity $\langle \alpha, v_j \rangle = \langle v_j, \alpha \rangle^* = \alpha_j^*$.

Upon the change of basis, the representations of vectors also change: Suppose that a vector $\alpha \in \mathcal{V}$ is represented by

$$\alpha \doteq \begin{bmatrix} \alpha_1 \\ \alpha_2 \\ \vdots \\ \alpha_n \end{bmatrix} \quad (\text{A.11})$$

in the basis $\{v_i\}$, and by

$$\alpha \doteq \begin{bmatrix} \alpha'_1 \\ \alpha'_2 \\ \vdots \\ \alpha'_n \end{bmatrix} \quad (\text{A.12})$$

in the basis $\{w_j\}$. Obviously, the relation between the two representations is fixed by the relation between the two bases. Let us take a closer look at the relation when the two bases are orthonormal: We note from (A.8) that

$$\alpha'_k = \langle w_k, \alpha \rangle = \sum_{j=1}^n \langle w_k, v_j \rangle \langle v_j, \alpha \rangle = \sum_k U_{kj} \alpha_j, \quad (\text{A.13})$$

where we have put $U_{kj} := \langle w_k, v_j \rangle$. We have seen in Eq. (A.5) that the matrix U is unitary. Therefore, we see that the representations in two different orthonormal bases are related by a unitary matrix as

$$\begin{bmatrix} \alpha'_1 \\ \alpha'_2 \\ \vdots \\ \alpha'_n \end{bmatrix} = \begin{bmatrix} U_{11} & U_{12} & \cdots & U_{1n} \\ U_{21} & U_{22} & \cdots & U_{2n} \\ \vdots & \vdots & \ddots & \vdots \\ U_{n1} & U_{n2} & \cdots & U_{nn} \end{bmatrix} \begin{bmatrix} \alpha_1 \\ \alpha_2 \\ \vdots \\ \alpha_n \end{bmatrix}. \quad (\text{A.14})$$

A.2 Linear Operators

In quantum mechanics, the evolution of quantum states and the properties of physical quantities are described by linear operators. Linear operators are special kind of linear mappings.

A.2.1 Linear Maps

As already mentioned, the most important algebraic property of a vector space is the superposition. Therefore, the mapping preserving this property from one vector space to another plays an important role in the theory of linear algebra.

Definition A.6 (*linear map*) Let \mathcal{V} and \mathcal{W} be vector spaces. A mapping (or simply map)

$$\hat{L} : \mathcal{V} \rightarrow \mathcal{W}, \quad v \mapsto \hat{L}v \quad (\text{A.15})$$

is said to be *linear* if it satisfies the following two properties:

- (a) For any $v, w \in \mathcal{V}$, we have $\hat{L}(v + w) = \hat{L}v + \hat{L}w$.
- (b) For all $z \in \mathbb{C}$ and $v \in \mathcal{V}$ we have $\hat{L}(zv) = z(\hat{L}v)$.

When $\mathcal{V} = \mathcal{W}$, the map is called a *linear operator* on \mathcal{V} .

As a linear map preserves superposition, it is completely determined by specifying how it maps just the basis vectors. The following theorem just summarizes the property.

Theorem A.7 Let \mathcal{V} and \mathcal{W} be vector spaces. Let $\{v_1, \dots, v_n\}$ be a basis of \mathcal{V} , and $w_1, \dots, w_n \in \mathcal{W}$ be arbitrary vectors—not to be necessarily distinctive nor to form a basis. Then there exists a *unique* linear map $\hat{L} : \mathcal{V} \rightarrow \mathcal{W}$ such that $w_j = \hat{L}v_j$ for all $j = 1, \dots, n$.

(proof) Define a map $\hat{A} : \mathcal{V} \rightarrow \mathcal{W}$ by the associations

$$\hat{A}v_j \mapsto w_j \quad (\dagger 1)$$

and

$$\hat{A}(v_1z_1 + \dots + v_nz_n) = w_1z_1 + \dots + w_nz_n \quad (\dagger 2)$$

for all $z_1, \dots, z_n \in \mathbb{C}$. Clearly \hat{A} is linear, and we have shown that there exists a linear map satisfying the required condition. Now suppose that two linear maps \hat{A} and \hat{B} satisfy the condition. Let $v = v_1z_1 + \dots + v_nz_n \in \mathcal{V}$ with $z_j \in \mathbb{C}$. Note that

$$\hat{B}v = (\hat{B}v_1)z_1 + \dots + (\hat{B}v_n)z_n = w_1z_1 + \dots + w_nz_n = \hat{A}v. \quad (\dagger 3)$$

As v is arbitrary, we conclude that $\hat{A} = \hat{B}$.

A.2.2 Representations

As asserted by Theorem A.7, a linear map $\hat{L} : \mathcal{V} \rightarrow \mathcal{W}$ is completely determined by specifying how it maps each element of a basis $\{v_j : j = 1, \dots, n\}$ of \mathcal{V} . Expanding the result $\hat{L}v_j$ in a basis $\{w_i : i = 1, \dots, m\}$ of \mathcal{W} as

$$\hat{L}v_j = \sum_i w_i L_{ij}, \quad (\text{A.16})$$

we can equivalently say that \hat{L} is *uniquely* specified by the coefficients $L_{ij} \in \mathbb{C}$. The $m \times n$ matrix composed of the coefficients is said to be the matrix representation of \hat{L} in the bases $\{v_j\}$ and $\{w_i\}$, and is denoted by

$$\hat{L} \doteq \begin{bmatrix} L_{11} & L_{12} & \cdots \\ L_{21} & L_{22} & \cdots \\ \vdots & \vdots & \ddots \end{bmatrix}. \quad (\text{A.17})$$

When the bases $\{v_j\}$ and $\{w_i\}$ are *orthonormal*, the matrix elements L_{ij} can be obtained by means of the Hermitian product, $L_{ij} = \langle w_i, \hat{L}v_j \rangle$, and hence

$$\hat{L}v_j = \sum_i w_i \langle w_i, \hat{L}v_j \rangle. \quad (\text{A.18})$$

In quantum mechanics, one has to calculate frequently the matrix representations of linear maps in orthonormal bases, and the procedure is summarized in the following table:

	v_1	v_2	\cdots
w_1	$\langle w_1, \hat{L}v_1 \rangle$	$\langle w_1, \hat{L}v_2 \rangle$	\cdots
w_2	$\langle w_2, \hat{L}v_1 \rangle$	$\langle w_2, \hat{L}v_2 \rangle$	\cdots
\vdots	\vdots	\vdots	\ddots

(A.19)

When $\mathcal{V} = \mathcal{W}$, a linear operator is represented by a square matrix. Let $\{v_i\}$ and $\{w_j\}$ are two different orthonormal bases of \mathcal{V} . As they are both orthonormal, there must be a unitary operator \hat{U} such that

$$w_j = \hat{U}v_j = \sum_i v_i U_{ij}, \quad (\text{A.20})$$

where $U_{ij} := \langle v_i, w_j \rangle$ is a unitary matrix—see Eq. (A.4). Suppose that L_{ij} be the matrix representation of a linear operator \hat{L} in the basis $\{v_i\}$. What is the matrix representation L'_{ij} of \hat{L} in the new basis $\{w_j\}$? As $\{w_j\}$ is orthonormal, the matrix representation is given by

$$L'_{ij} = \langle w_i, \hat{L}w_j \rangle = \sum_{kl} U_{ik}^* \langle v_i, \hat{L}v_l \rangle U_{lj} = \sum_{kl} U_{ik}^* L_{il} U_{lj}, \quad (\text{A.21})$$

that is, the matrix representations in two different bases are related with each other by

$$L' = U^\dagger L U. \quad (\text{A.22})$$

A.2.3 Hermitian Conjugate of Operators

On a vector space equipped with a Hermitian product, given a linear operator \hat{L} one can define another linear operator \hat{L}^\dagger naturally related to \hat{L} . Hermitian conjugates of operators greatly simplify the evaluations of operator-related expressions and spectral analysis of them.

Theorem A.8 (Hermitian conjugate) Let \mathcal{V} and \mathcal{W} be vector spaces over \mathbb{C} that are equipped with Hermitian products. Let $\hat{L} : \mathcal{V} \rightarrow \mathcal{W}$ be a linear map. Then, the following statements hold true:

- (a) There exists a *unique* linear map $\hat{L}^\dagger : \mathcal{W} \rightarrow \mathcal{V}$ such that

$$\langle w, \hat{L}v \rangle_{\mathcal{W}} = \langle \hat{L}^\dagger w, v \rangle_{\mathcal{V}} \quad (\text{A.23})$$

for all $v \in \mathcal{V}$ and $w \in \mathcal{W}$.

- (b) $(\hat{L}^\dagger)^\dagger$ exists as well, and is identical to \hat{L} , $(\hat{L}^\dagger)^\dagger = \hat{L}$.

The linear operator \hat{L}^\dagger is called the *Hermitian conjugate* of \hat{L} .

As the matrix representation of a linear map is unique, one can also define the Hermitian conjugate in terms of the matrix representation. Let $\hat{L} : \mathcal{V} \rightarrow \mathcal{W}$ be represented by

$$\hat{L}v_j = \sum_i w_i L_{ij}. \quad (\text{A.24})$$

Then, $\hat{L}^\dagger : \mathcal{W} \rightarrow \mathcal{V}$ is defined by

$$\hat{L}^\dagger w_j = \sum_i v_i L_{ji}^*. \quad (\text{A.25})$$

That is, the matrix representation of \hat{L}^\dagger is the conjugate-transpose of the matrix representation of \hat{L} . For finite-dimensional vector spaces, the two definitions are equivalent.

For an operator on a vector space, its Hermitian conjugate also acts on the same vector space, and enables to characterize the operator itself.

Definition A.9 (normal operator) A linear operator \hat{L} on a vector space \mathcal{V} is said to be *normal* if $[\hat{L}^\dagger, \hat{L}] = 0$.

The two most important examples of normal operators are Hermitian operators and unitary operators.

In quantum mechanics, the linear operator representing a physical quantity should be Hermitian:

Definition A.10 (*Hermitian operator*) A linear operator \hat{H} on a vector space is called *Hermitian* if

$$\langle \hat{H}v, w \rangle = \langle v, \hat{H}w \rangle , \quad \forall v, w \in \mathcal{V}. \quad (\text{A.26})$$

There is a simple test for a Hermitian operator on a finite dimensional vector space:

Theorem A.11 Let \mathcal{V} be a vector space. An operator \hat{H} on \mathcal{V} is *Hermitian* if and only if $\langle v, \hat{H}v \rangle \in \mathbb{R}$ for all $v \in \mathcal{V}$.

In quantum mechanics, operators usually describe the changes of state of a system by means of transformation of vectors. However, there is a special example where the operator itself describes the “state” of the system. That is, the density operator describes the mixed state of the system. For the proper statistical interpretation of the mixed state, density operators are required to satisfy certain properties. They are Hermitian and positive among others properties. It motivates the following definition.

Definition A.12 (*positive operator*) A Hermitian operator \hat{H} on a vector space \mathcal{V} is said to be *positive* (or more specifically, *positive definite*) if $\langle v, \hat{H}v \rangle > 0$ for all $v \in \mathcal{V}$ ($v \neq 0$). A positive operator is denoted as $\hat{H} > 0$. It is said to be *positive semidefinite* or *non-negative* if $\langle v, \hat{H}v \rangle \geq 0$ for all $v \in \mathcal{V}$ ($v \neq 0$). It is denoted as $\hat{H} \geq 0$.

Another kind of operators one can encounter very frequently in quantum mechanics is unitary operators, *norm-preserving and invertible* linear maps.

Definition A.13 Let \mathcal{V} be a vector space \mathcal{V} equipped with a Hermitian product. A linear operator \hat{U} is said to be *unitary* when it maps \mathcal{V} onto the whole of \mathcal{V} and preserve the norm. That is, $\hat{U}\mathcal{V} = \mathcal{V}$ and $\langle \hat{U}v, \hat{U}v \rangle = \langle v, v \rangle$ for all $v \in \mathcal{V}$.

A unitary operator is characterized by the fact that its Hermitian conjugate is identical to its inverse.

Theorem A.14 If \hat{U} is a unitary operator on \mathcal{V} , then

$$\hat{U}^\dagger \hat{U} = \hat{U} \hat{U}^\dagger = 1. \quad (\text{A.27})$$

In fact, Eq. (A.27) can be used as an alternative definition of a unitary operator.

The unique linear map in Theorem A.7 becomes a unitary operator when the image vectors form another orthonormal basis of the same vector space.

Theorem A.15 Let \mathcal{V} be a vector space. Let $\{v_1, \dots, v_n\}$ and $\{w_1, \dots, w_n\}$ be *orthonormal* bases of \mathcal{V} . Then there exists a *unique* unitary operator \hat{U} on \mathcal{V} such that $w_j = \hat{U}v_j$ for all $j = 1, \dots, n$.

We have already seen in Eq. (A.4) that two orthonormal bases are related by a unitary matrix. Theorem A.15 just asserts it again. Indeed, if U is the matrix representation of \hat{U} in the basis $\{v_i\}$, then

$$w_j = \hat{U}v_j = \sum_i v_i U_{ij}. \quad (\text{A.28})$$

Theorem A.15 is even more general and hold for any orthonormal subsets:

Theorem A.16 Let \mathcal{V} is a Hilbert space equipped with a *positive-definite* Hermitian product $\langle \cdot, \cdot \rangle$, and $\mathcal{U} \subset \mathcal{V}$ a subspace. Suppose $\hat{U} : \mathcal{U} \rightarrow \mathcal{V}$ is a linear operator which preserves the Hermitian product. That is, for any $u, u' \in \mathcal{U}$,

$$\langle \hat{U}u, \hat{U}u' \rangle = \langle u, u' \rangle. \quad (\text{A.29})$$

There exists a unitary operator $\hat{V} : \mathcal{V} \rightarrow \mathcal{V}$ which extends \hat{U} . That is, $\hat{V}u = \hat{U}u$ for all $u \in \mathcal{U}$ and \hat{V} is defined on the entire space \mathcal{V} .

An immediate consequence of Theorem A.16 is that for any pair of vectors v and w one can always find a unitary operator \hat{U} such that $w = \hat{U}v$.

A.3 Dirac's Bra-Ket Notation

For a given vector space \mathcal{V} , one can construct another special vector space \mathcal{V}^* associated with \mathcal{V} , consisting of all linear mappings from \mathcal{V} to \mathbb{C} , $\mathcal{V}^* := \{\phi : \mathcal{V} \rightarrow \mathbb{C}\}$. It is called the *dual space* of \mathcal{V} . With a fixed vector $v \in \mathcal{V}$ and the Hermitian product, one can define a linear mapping $\phi_v : \mathcal{V} \rightarrow \mathbb{C}$ by the relation $\phi_v(w) := \langle v, w \rangle$. Certainly, ϕ_v is an element of \mathcal{V}^* . This way, by choosing different vectors from \mathcal{V} , one can define a particular kind of linear mappings belonging to \mathcal{V}^* . Now the key observation is that in fact, any linear mapping in \mathcal{V}^* is of this kind. That is, there is a one-to-one correspondence $v \leftrightarrow \phi_v$ between \mathcal{V} and \mathcal{V}^* . ϕ_v is called the *dual vector* of v .

In Dirac's bra-ket notation, the dual ϕ_v is denoted by $\langle v |$ whereas the native vector v is denoted by $|v\rangle$; hence the name. It is just a simple notational change. However, it simplifies most evaluations in quantum mechanics so greatly that it is widely used. ***Throughout the book, we will almost always be using the bra-ket notation.***

When $\{| \alpha_1 \rangle, \dots, | \alpha_n \rangle\}$ is an orthonormal basis of \mathcal{V} , $\{ \langle \alpha_1 |, \dots, \langle \alpha_n | \}$ is also an orthonormal basis of \mathcal{V}^* and called the *dual basis* of the former. More importantly, the two bases satisfy the relation

$$\langle \alpha_i | \alpha_j \rangle = \delta_{ij}. \quad (\text{A.30})$$

Suppose that a vector $|v\rangle \in \mathcal{V}$ is expanded as

$$|v\rangle = \sum_j |\alpha_j\rangle v_j, \quad v_j \in \mathbb{C}. \quad (\text{A.31})$$

Then, its dual vector $\langle v|$ is given by

$$\langle v| = \sum_j v_j^* \langle \alpha_j|. \quad (\text{A.32})$$

Armed with the basic properties of the bra-ket notation, now consider a combination of the form $|v\rangle \langle v'|$, where $|v\rangle, |v'\rangle \in \mathcal{V}$: We regard it as an operator on \mathcal{V} defined by the association

$$|v\rangle \langle v'| : |u\rangle \mapsto |v\rangle \langle v'|u\rangle \quad (\text{A.33})$$

for all $|u\rangle \in \mathcal{V}$. A simple inspection—see Definition A.8—shows that its Hermitian conjugate $(|v\rangle \langle v'|)^\dagger$ is just given by

$$(|v\rangle \langle v'|)^\dagger = |v'\rangle \langle v|. \quad (\text{A.34})$$

If one constructs $|\alpha_i\rangle \langle \alpha_j|$ out of a basis $\{|\alpha_j\rangle\}$ of \mathcal{V} , then any linear map on \mathcal{V} can be expressed in terms of them—they form a basis of the vector space $\mathcal{L}(\mathcal{V})$ of linear operators (Appendix B.1). In particular, if the basis $\{|\alpha_j\rangle\}$ is *orthonormal*, then a linear operator \hat{L} on \mathcal{V} with the matrix representation L in the same basis is equivalent to

$$\hat{L} = \sum_{ij} |\alpha_i\rangle L_{ij} \langle \alpha_j|, \quad (\text{A.35})$$

and, accordingly, its Hermitian conjugate \hat{L}^\dagger to

$$\hat{L}^\dagger = \sum_{ij} |\alpha_i\rangle L_{ji}^* \langle \alpha_j|. \quad (\text{A.36})$$

Both expansions can be verified by evaluating the matrix elements in the basis and applying the orthogonality relation (A.30). An interesting result is that for *any* orthonormal basis $\{|\alpha_j\rangle\}$ of \mathcal{V} , the linear combination

$$\hat{I} = \sum_j |\alpha_j\rangle \langle \alpha_j| \quad (\text{A.37})$$

is equal to the identity operator on \mathcal{V} . It is called the *completeness relation*. The orthogonality relation (A.30) and the completeness relation (A.37), which are mutually complementary, together empower the bra-ket notation.

Consider a system of two qubits. The associated Hilbert space is spanned by the standard basis.

```
In[7]:= bs = Basis[2]
Out[7]= { |0, 0⟩, |0, 1⟩, |1, 0⟩, |1, 1⟩}
```

Consider a Hermitian operator, physically, corresponding to the Heisenberg exchange interaction between two S=1/2 spins.

```
In[8]:= op = Pauli[1, 1] + Pauli[2, 2] + Pauli[3, 3]
Out[8]= σx ⊗ σx + σy ⊗ σy + σz ⊗ σz
```

This shows the matrix representation of the operator.

```
In[9]:= mat = Matrix[op];
mat // MatrixForm
Out[9]/MatrixForm=

```

$$\begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & -1 & 2 & 0 \\ 0 & 2 & -1 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix}$$

This is an expansion of the operator in the bra-ket notation.

```
In[10]:= op2 = MultiplyDot[bs, mat, Dagger[bs]]
Out[10]= |0, 0⟩⟨0, 0| - |0, 1⟩⟨0, 1| + 2 |0, 1⟩⟨1, 0| +
2 |1, 0⟩⟨0, 1| - |1, 0⟩⟨1, 0| + |1, 1⟩⟨1, 1|
```

Verify the above expansion by evaluating the matrix representation.

```
In[11]:= mat2 = Outer[Multiply, Dagger[bs], op2 ** bs];
mat2 // MatrixForm
Out[11]/MatrixForm=

```

$$\begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & -1 & 2 & 0 \\ 0 & 2 & -1 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix}$$

More generally, given two vector spaces \mathcal{V} and \mathcal{W} , the combination $|w\rangle\langle v|$ with for $|v\rangle \in \mathcal{V}$ and $|w\rangle \in \mathcal{W}$ is a linear map $\mathcal{V} \rightarrow \mathcal{W}$ with an association similar to that in Eq. (A.33). Its Hermitian conjugate, $(|w\rangle\langle v|)^\dagger : \mathcal{W} \rightarrow \mathcal{V}$, is given by $(|w\rangle\langle v|)^\dagger = |v\rangle\langle w|$.

To illustrate the power of the bra-ket notation, let us consider a linear map $\hat{L} : \mathcal{V} \rightarrow \mathcal{W}$ and examine $\hat{L}|_{\alpha_j}\rangle$: Let $\{\alpha_j\} : j = 1, \dots, m\}$ and $\{\beta_k\} : k = 1, \dots, n\}$ be orthonormal bases of \mathcal{V} and \mathcal{W} , respectively. As $\hat{L}|_{\alpha_j}\rangle$ is an element of \mathcal{W} , it should not be affected by the identity operator $\hat{I}_{\mathcal{W}}$ on \mathcal{W} , $\hat{L}|_{\alpha_j}\rangle = \hat{I}_{\mathcal{W}}\hat{L}|_{\alpha_j}\rangle$. Using the completeness relation (A.37) (replacing $|\alpha_j\rangle$ with $|\beta_k\rangle$), one can get

$$\hat{L}|_{\alpha_j}\rangle = \hat{I}_{\mathcal{W}}\hat{L}|_{\alpha_j}\rangle = \sum_k |\beta_k\rangle\langle\beta_k| \hat{L}|_{\alpha_j}\rangle. \quad (\text{A.38})$$

Noting that $L_{kj} = \langle\beta_k|\hat{L}|_{\alpha_j}\rangle$ is the matrix elements of the representation of \hat{L} in the given bases [see Eq. (A.18)], one recovers the defining relation [see Eq. (A.16)]

$$\hat{L}|_{\alpha_j}\rangle = \sum_k |\beta_k\rangle L_{kj} \quad (\text{A.39})$$

for the matrix representation of \hat{L} . Given the matrix representation L_{kj} , one can further expand \hat{L} in terms of the bra-ket notation as

$$\hat{L} = \sum_{kj} |\beta_k\rangle L_{kj} \langle \alpha_j| . \quad (\text{A.40})$$

Once expanded in the bra-ket notation, its Hermitian conjugate $\hat{L}^\dagger : \mathcal{W} \rightarrow \mathcal{V}$ reads as

$$\hat{L}^\dagger = \sum_{kj} |\alpha_j\rangle L_{kj}^* \langle \beta_k| . \quad (\text{A.41})$$

A.4 Spectral Theorems

The eigenvalues and eigenvectors of normal operators—see Definition A.9—exhibit particularly useful properties. They are frequently used in quantum mechanics and simplifies many calculations and analyses. Here we summarize the properties of eigenvalues and eigenvectors of normal operators, especially, Hermitian and unitary operators. Although we will be focusing on the spectral properties, we will also discuss some other related properties.

A.4.1 Spectral Decomposition

The following theorems summarize the properties of the eigenvectors and eigenvalues of normal operators, especially, Hermitian, positive, and unitary operators:

Theorem A.17 Let \hat{A} be a *normal operator* (see Definition A.9) on a vector space \mathcal{V} .

- (a) Eigenstates of \hat{A} belonging to distinct eigenvalues are orthogonal to each other.
- (b) The set of all eigenvectors of \hat{A} spans \mathcal{V} .

Theorem A.18 (a) A Hermitian operator \hat{H} is normal. That is, eigenvectors belonging to different eigenvalues are orthogonal to each other.

- (b) Every eigenvalues of a Hermitian operator is real.

Theorem A.19 Let \hat{H} be a Hermitian operator on a vector space. Then,

- (a) \hat{H} is positive if and only if every eigenvalue of it is positive;
- (b) \hat{H} is positive-semidefinite if and only if the eigenvalues are non-negative.

Theorem A.20 Let \hat{U} be a unitary operator on a vector space \mathcal{V} . Then, every eigenvalue of \hat{U} is of the form $e^{i\phi}$ with $\phi \in \mathbb{R}$.

Theorem A.17 enables to expand a normal operator in terms of its eigenvectors and eigenvalues using Dirac's bra-ket notation: Suppose that a normal operator \hat{A} on a vector space \mathcal{V} has eigenvectors $|a\rangle$ and the corresponding eigenvalues a . If some eigenvalues are degenerate, that is, there are more than one linearly independent eigenvectors belonging to the same eigenvalue, we choose mutually orthogonal eigenvectors—this is always possible. Then, the normalized eigenvectors form an orthonormal basis, which is called the *eigenbasis* from \hat{A} of the vector space \mathcal{V} . Then the matrix representation of \hat{A} in the eigenbasis from itself must be diagonal with the diagonal elements given by the eigenvalues. Hence, in Dirac's bra-ket notation, \hat{A} can be expanded as

$$\hat{A} = \sum_a |a\rangle a \langle a| , \quad (\text{A.42})$$

where the sum is over all the eigenvalues of \hat{A} . The expansion is called the *spectral decomposition* of \hat{A} .

Sometimes, it is useful to normalize the eigenvectors $|a\rangle$ of a *positive operator* (see Definition A.12 and Theorem A.19) by their own (positive) eigenvalues a so that $\langle a|a\rangle = a$. In this case, the spectral decomposition of a positive operator \hat{A} is given by

$$\hat{A} = \sum_a |a\rangle \langle a| , \quad (\text{A.43})$$

This form of the spectral decomposition for a positive operator should not be confused with the completeness relation (A.37), where an orthonormal basis is used. For a *positive semidefinite operator*, there only appear eigenvectors with positive eigenvalues in the summation in (A.43)—the eigenvectors with zero eigenvalue are dropped automatically.

A.4.2 Functions of Operators

The spectral decomposition provides a convenient way to define *functions of a normal operator*. Let $f : \mathcal{D} \rightarrow \mathbb{C}$ be a function of complex variable defined in a domain $\mathcal{D} \subset \mathbb{C}$. Suppose that \hat{A} be a normal operator with all eigenvalues $a \in \mathcal{D}$. Then the function $f(\hat{A})$ of the operator \hat{A} —another operator on the same vector space derived from \hat{A} —is defined by [to be compared with (A.42)]

$$f(\hat{A}) := \sum_a |a\rangle f(a) \langle a| . \quad (\text{A.44})$$

Surprisingly, most students try to define a function of an operator by means of Taylor series expansion which involves multiple powers of the matrix. In most cases, however, it is very difficult to convince oneself whether the series is actually converging or not. Even if it does, it is tremendously difficult to figure out the behavior of the resulting operator, not to speak of the evaluation of the series itself. In contrast, the definition in (A.44) is well-defined as long as $f(z)$ of complex

variable z is well defined, and often provides clear physical meaning of the resulting operator $f(\hat{A})$. Above all, the evaluation is straightforward and, in many cases, simple. The definition applies only to normal operators. However, it is not a serious restriction since in most physics applications it is normal operators that we want to transform by means of already existing functions.

Consider again a Hermitian operator describing the Heisenberg exchange interaction between two $S=1/2$ spins.

```
In[1]:= opH = -J * (Pauli[1, 1] + Pauli[2, 2] + Pauli[3, 3])
```

```
Out[1]= -J (σx ⊗ σx + σy ⊗ σy + σz ⊗ σz)
```

We want to consider functions of operators, for example, `opH` in particular. To do it, it is most efficient to proceed with the spectral decomposition of the operator.

```
In[2]:= {val, vec} = ProperSystem[opH]
```

```
Out[2]= {{3 J, -J, -J, -J}, {-|0, 1⟩ + |1, 0⟩, |1, 1⟩, |0, 1⟩ + |1, 0⟩, |0, 0⟩}}
```

The eigenvectors are orthogonal, but not properly normalized.

```
In[3]:= Outer[Multiply, Dagger[vec], vec] // MatrixForm
```

```
Out[3]/MatrixForm=
```

$$\begin{pmatrix} 2 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 2 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix}$$

Normalize them, and check again.

```
In[4]:= nvec = vec / Sqrt[{2, 1, 2, 1}]
Outer[Multiply, Dagger[nvec], nvec] // MatrixForm
```

```
Out[4]= { -|0, 1⟩ + |1, 0⟩ / √2, |1, 1⟩, |0, 1⟩ + |1, 0⟩ / √2, |0, 0⟩ }
```

```
Out[4]/MatrixForm=
```

$$\begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix}$$

Suppose we want to get the exponential function of `opH`. For example, the time-evolution operator is given by

```
In[5]:= opU = MultiplyExp[-I t opH]
```

```
Out[5]= ei J t (σx ⊗ σx + σy ⊗ σy + σz ⊗ σz)
```

This uses the spectral decomposition.

```
In[6]:= newU = Total@Multiply[nvec, Exp[-I t val], Dagger[nvec]]
```

```
Out[6]= ei J t |0, 0⟩ ⟨0, 0| + 1/2 ei J t |0, 1⟩ ⟨0, 1| +
1/2 e-i J t |0, 1⟩ ⟨0, 1| + 1/2 ei J t |0, 1⟩ ⟨1, 0| -
1/2 e-i J t |0, 1⟩ ⟨1, 0| + 1/2 ei J t |1, 0⟩ ⟨0, 1| - 1/2 e-i J t |1, 0⟩ ⟨0, 1| +
1/2 ei J t |1, 0⟩ ⟨1, 0| + 1/2 e-i J t |1, 0⟩ ⟨1, 0| + ei J t |1, 1⟩ ⟨1, 1|
```

This converts the bra-ket expression into a form in terms of the Pauli operators.

```
In[7]:= newU2 = Elaborate@ExpressionFor@Matrix[newU]
Out[7]= 
$$\frac{1}{4} e^{-3i\sqrt{3}t} (1 + 3e^{4i\sqrt{3}t}) \sigma^0 \otimes \sigma^0 + \frac{1}{4} e^{-3i\sqrt{3}t} (-1 + e^{4i\sqrt{3}t}) \sigma^x \otimes \sigma^x +$$


$$\frac{1}{4} e^{-3i\sqrt{3}t} (-1 + e^{4i\sqrt{3}t}) \sigma^y \otimes \sigma^y + \frac{1}{4} e^{-3i\sqrt{3}t} (-1 + e^{4i\sqrt{3}t}) \sigma^z \otimes \sigma^z$$

```

In many cases, `MultiplyExp` can be further evaluated by means of `Elaborate`.

```
In[8]:= opU2 = Elaborate@Elaborate[opU]
Out[8]= 
$$\frac{1}{4} e^{-3i\sqrt{3}t} (1 + 3e^{4i\sqrt{3}t}) \sigma^0 \otimes \sigma^0 + \frac{1}{4} e^{-3i\sqrt{3}t} (-1 + e^{4i\sqrt{3}t}) \sigma^x \otimes \sigma^x +$$


$$\frac{1}{4} e^{-3i\sqrt{3}t} (-1 + e^{4i\sqrt{3}t}) \sigma^y \otimes \sigma^y + \frac{1}{4} e^{-3i\sqrt{3}t} (-1 + e^{4i\sqrt{3}t}) \sigma^z \otimes \sigma^z$$

```

A.5 Tensor-Product Spaces

When there are more than one systems, each system is associated with a different vector space. Even for a single system, independent degrees of freedom (such as external and internal degrees of freedom) are associated with different vector spaces. Then the vector space of the total system should be constructed by the vector spaces associated with the individual systems or degrees of freedom. Mathematically, such a construction is called the tensor product of the constituent vector spaces.

A.5.1 Vectors in a Product Space

Let \mathcal{V} and \mathcal{W} be vector spaces, and $\{|v_i\rangle\}$ and $\{|w_j\rangle\}$ their respective bases. The tensor product $\mathcal{V} \otimes \mathcal{W}$ is a vector space spanned by the basis

$$\{|v_i\rangle \otimes |w_j\rangle : i = 1, \dots, \dim \mathcal{V}; j = 1, \dots, \dim \mathcal{W}\} \quad (\text{A.45})$$

We call it the *standard basis* of $\mathcal{V} \otimes \mathcal{W}$. Obviously, the dimension of $\mathcal{V} \otimes \mathcal{W}$ is given by $\dim(\mathcal{V} \otimes \mathcal{W}) = (\dim \mathcal{V}) \times (\dim \mathcal{W})$. The symbol ‘ \otimes ’ in the basis states separates $|v_i\rangle$ and $|w_j\rangle$ from each other clearly indicating which vector space they are from. In many cases, when there is no risk of confusion, it is dropped and the product states are simply written as $|v_i\rangle |w_j\rangle$ or even $|v_i w_j\rangle$.

Here is the standard basis (product basis) for a (unlabelled) two-qubit system.

```
In[9]:= bs = Basis[2];
bs // LogicalForm
Out[9]= {|0, 0>, |0, 1>, |1, 0>, |1, 1>}
```

The Hermitian products $\langle \cdot, \cdot \rangle_{\mathcal{V}}$ and $\langle \cdot, \cdot \rangle_{\mathcal{W}}$ in the corresponding spaces are inherited to the tensor-product space to give the standard Hermitian product

$$(\langle v_i | \otimes \langle w_j |)(|v_k\rangle \otimes |w_l\rangle) = \langle v_i | v_k \rangle_{\mathcal{V}} \langle w_j | w_l \rangle_{\mathcal{W}}. \quad (\text{A.46})$$

Expanded in the standard basis, a vector

$$|\Psi\rangle = \sum_{ij} |v_i\rangle \otimes |w_j\rangle \Psi_{ij} \in \mathcal{V} \times \mathcal{W} \quad (\text{A.47})$$

involves $M \times N$ terms in general, where $M := \dim \mathcal{V}$ and $N := \dim \mathcal{W}$. It can be reduced to a form with much less terms. To see this, rewrite the matrix Ψ consisting of the expansion coefficients Ψ_{ij} by means of the singular-value decomposition

$$\Psi = U \Sigma V^\dagger, \quad (\text{A.48})$$

where U is a $M \times M$ unitary matrix, V a $N \times N$ matrix, and Σ a $M \times N$ diagonal matrix. The diagonal elements s_j of Σ are all non-negative, and the number R of non-zero elements cannot be greater than $\min(M, N)$. Putting (A.48) back into (A.47) leads to the so-called *Schmidt decomposition*

$$|\Psi\rangle = \sum_{j=1}^R |\alpha_j\rangle \otimes |\beta_j\rangle s_j, \quad |\alpha_j\rangle := \sum_i |v_i\rangle U_{ij}, \quad |\beta_j\rangle := \sum_i |w_i\rangle V_{ij}^*. \quad (\text{A.49})$$

Note that $\langle \alpha_i | \alpha_j \rangle = \delta_{ij}$ and $\langle \beta_i | \beta_j \rangle = \delta_{ij}$ because U and V are unitary, and that $\sum_{j=1}^R s_j^2 = 1$ ($0 < s_j < 1$) if $|\Psi\rangle$ is normalized.

The number R is called the *Schmidt rank* or *Schmidt number* of the vector $|\Psi\rangle$. When $R = 1$, $|\Psi\rangle$ is factorized as $|\Psi\rangle = |\alpha_1\rangle \otimes |\beta_1\rangle$, and is said to be *separable*. Otherwise, it cannot be factorized and it is called an *entangled vector*. The Schmidt decomposition is a convenient method to test whether a vector is separable or entangled.

Consider an arbitrary vector in the tensor-product space.

```
In[7]:= cc = Re@RandomVector[4];
vec = bs.cc;
vec // LogicalForm
Out[7]= -0.392437 |0, 0⟩ + 0.309225 |0, 1⟩ - 0.352869 |1, 0⟩ + 0.733405 |1, 1⟩
```

This is its Schmidt decomposition and shows that the state vector is entangled.

```
In[8]:= {ww, uu, vv} = SchmidtDecomposition[vec, {1}, {2}];
ww // Normal
uu
vv
Out[8]= {0.935711, 0.190978}
Out[9]= {0.504017 |0⟩ + 0.863694 |1⟩, -0.863694 |0⟩ + 0.504017 |1⟩}
Out[10]= {-0.537096 |0⟩ + 0.843521 |1⟩, 0.843521 |0⟩ + 0.537096 |1⟩}
```

`SchmidtForm` presents the Schmidt decomposition in a more intuitively-appealing form. For a thorough analysis of the result, use `SchmidtDecomposition`.

```
In[7]:= new = SchmidtForm[vec, {1}, {2}]
Out[7]= 0.190978 (-0.863694 |0⟩ + 0.504017 |1⟩) ⊗ (0.843521 |0⟩ + 0.537096 |1⟩) +
         0.935711 (0.504017 |0⟩ + 0.863694 |1⟩) ⊗ (-0.537096 |0⟩ + 0.843521 |1⟩)
```

Check whether the two vectors are the same or not.

```
In[8]:= vec - ReleaseTimes[new] // Garner // Chop
Out[8]= 0
```

A.5.2 Operators on a Product Space

Let \hat{A} and \hat{B} be linear operators on \mathcal{V} and \mathcal{W} , respectively. Then the *tensor-product operator* $\hat{A} \otimes \hat{B}$ is a linear operator on the tensor-product space $\mathcal{V} \otimes \mathcal{W}$ defined by the association

$$(\hat{A} \otimes \hat{B})(|v_i\rangle \otimes |w_j\rangle) = (\hat{A}|v_i\rangle) \otimes (\hat{B}|w_j\rangle). \quad (\text{A.50})$$

Suppose that \hat{A} and \hat{B} are represented by matrices A and B , respectively, i.e.,

$$\hat{A}|v_j\rangle = \sum_i |v_i\rangle A_{ij}, \quad \hat{B}|v_j\rangle = \sum_i |w_i\rangle B_{ij}. \quad (\text{A.51})$$

Then it follows from (A.50) that

$$(\hat{A} \otimes \hat{B})(|v_i\rangle \otimes |w_j\rangle) = \left(\sum_k |v_k\rangle A_{ki} \right) \otimes \left(\sum_l |w_l\rangle B_{lj} \right) = \sum_{kl} |v_k\rangle \otimes |w_l\rangle A_{ki} B_{lj}. \quad (\text{A.52})$$

It implies that the matrix representation of $\hat{A} \otimes \hat{B}$ in the standard product basis is given by the *direct product* $A \otimes B$ of the two matrices.

In general, a linear operator \hat{C} on $\mathcal{V} \otimes \mathcal{W}$ is not a single product but a sum of such products. How many terms are there? Suppose that the matrix $C_{ij;kl}$ is the matrix representation of \hat{C} in the standard product basis,

$$\hat{C}|v_kw_l\rangle = \sum_{ij} |v_iv_j\rangle C_{ij;kl}, \quad (\text{A.53})$$

or equivalently,

$$\hat{C} = \sum_{ij;kl} |v_iw_j\rangle \langle v_kw_l| C_{ij;kl} = \sum_{ij;kl} |v_i\rangle \langle v_k| \otimes |w_j\rangle \langle w_l| C_{ij;kl}. \quad (\text{A.54})$$

The $M^2 \times N^2$ matrix $G_{ik;jl} := C_{ij;kl}$ with collective indices (ik) and (jl) has a singular value decomposition

$$G_{ik;jl} = \sum_\mu U_{ik;\mu} \gamma_\mu V_{\mu;jl}^\dagger, \quad \gamma_\mu \geq 0. \quad (\text{A.55})$$

Defining

$$\hat{A}_\mu := \sum_{ik} |v_i\rangle \langle v_k| U_{ik;\mu}, \quad \hat{B}_\mu := \sum_{jl} |w_j\rangle \langle w_l| V_{jl;\mu}^*, \quad (\text{A.56})$$

leads to the expression

$$\hat{C} = \sum_\mu \hat{A}_\mu \otimes \hat{B}_\mu \gamma_\mu, \quad (\text{A.57})$$

which is in direct analogy with the Schmidt decomposition (A.49) of a vector in the tensor-product space. Here the number of non-vanishing singular values γ_μ is less than or equal to $\min(M^2, N^2)$.

Appendix B

Superoperators

• July 25, 2021 (v1.8)

A superoperator is a linear operator acting on a vector space of linear operators. As the concept of vectors is completely general, at a first glance there seems to be no reason why one should reserve a distinctive name for such operators and devote additional discussions. A considerable amount of interest in superoperators came with the booming of quantum information theory in the 1990s when it became clear that superoperators are important in the study of entanglement. Since then, mathematical theories on superoperators have been developed at a notably fast pace and applied to a wide range of subjects in quantum computation and quantum information. In this appendix, we briefly survey the properties of superoperators and provide some mathematical tools for the studies of entanglement and decoherence (see Chapter 5).

B.1 Operators as Vectors

The addition of two operators acting on a vector space as well as the multiplication of an operator by a scalar are defined in a natural and straightforward way. That is, operators on a vector space form a vector space themselves. Vector space of operators is not merely a mathematical generalization, but has an important physical relevance. In quantum physics, a mixed state, a statistical mixture of pure quantum states, is described by a so-called density operator.

There is another rather mathematically motivated and yet physically important fact that makes regarding operators as vectors very useful: Any unitary operator \hat{U} can be written in the form $\hat{U} = \exp(i\hat{H})$, where \hat{H} is an Hermitian operator. To describe any physical process, one has to deal with unitary operators. Because of the defining constraint, $\hat{U}^\dagger \hat{U} = \hat{I}$, it is often difficult to directly handle unitary operators. In most case, it is much more convenient and easier to handle Hermitian operators and consider the exponential function of them. As there is no constraint—apart from the rather trivial Hermiticity condition—for Hermitian operators, it is

natural to express them as linear combinations of some basis elements. It makes the handling of physically relevant operators much more tractable.

In this appendix, we will discuss the general structure of the vector space of all linear operators on a vector space. Before we discuss more general vector spaces of linear operators, let us first consider vector spaces of matrices.

Exercise B.1 (*matrices as vectors*) Consider the set \mathcal{M}_n of all $n \times n$ complex matrices.

- (a) Show that \mathcal{M}_n is a vector space.
- (b) What is the dimension of \mathcal{M}_n .
- (c) Define a *Hermitian product* in \mathcal{M}_n .
- (d) Construct an *orthogonal basis* of \mathcal{M}_n .

Here is an even more specific example.

Example B.2 (*Pauli decomposition*) Consider \mathcal{M}_2 .

- (a) Show that the four Pauli matrices $\hat{\sigma}^\mu$ ($\mu = 0, 1, 2, 3$) form an orthogonal basis.
- (b) Given an arbitrary matrix $L \in \mathcal{M}_2$, expand it in terms of the Pauli matrices. That is, find the most general form of L in terms of the Pauli matrices.
- (c) Find the most general form of a Hermitian matrix $H \in \mathcal{M}_2$.
- (d) Find the most general form of a unitary matrix $U \in \mathcal{M}_2$.

Let us demonstrate that any 2×2 matrix can be written as a linear superposition of the Pauli matrices. Consider an arbitrary 2×2 matrix.

```
In[1]:= L = {{1, 2 I},  
           {-I, 3}};  
L // MatrixForm
```

$$\begin{pmatrix} 1 & 2i \\ -i & 3 \end{pmatrix}$$

ExpressionFor converts a matrix into an operator expression in terms of the Pauli operators -- the Pauli operators are the operator forms of the Pauli matrices.

```
In[2]:= op = ExpressionFor[L]  
Elaborate[op]
```

$$\text{Out}[2]= 2 \sigma^0 - \sigma^z + 2 i \sigma^+ - i \sigma^-$$

$$\text{Out}[3]= 2 \sigma^0 + \frac{i \sigma^x}{2} - \frac{3 \sigma^y}{2} - \sigma^z$$

The symbols σ^μ in the above are the displayed form of Pauli.

```
In[4]:= InputForm[op]
```

$$\text{Out}[4]= 2 \text{Pauli}[0] - \text{Pauli}[3] + (2*I)*\text{Pauli}[4] - I*\text{Pauli}[5]$$

ThePauli is the matrix form of Pauli. The following statement reconstructs the original matrix.

```
In[1]:= new = 2 ThePauli[0] + (I / 2) ThePauli[1] - (3 / 2) ThePauli[2] - ThePauli[3];
new // MatrixForm
Out[1]//MatrixForm=

$$\begin{pmatrix} 1 & 2i \\ -i & 3 \end{pmatrix}$$

```

The conversion of Pauli to the corresponding matrix -- ThePauli -- can be achieved by simply using Matrix.

```
In[2]:= new2 = Matrix[op];
new2 // MatrixForm
Out[2]//MatrixForm=

$$\begin{pmatrix} 1 & 2i \\ -i & 3 \end{pmatrix}$$

```

Let us analyse the above demonstration in more detail. Here are the Pauli matrices. They form a basis of \mathcal{M}_2 .

```
In[3]:= bs = ThePauli /* {0, 1, 2, 3};
MatrixForm/@bs
Out[3]= \{\begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix}, \begin{pmatrix} 0 & 1 \\ 1 & 0 \end{pmatrix}, \begin{pmatrix} 0 & -i \\ i & 0 \end{pmatrix}, \begin{pmatrix} 1 & 0 \\ 0 & -1 \end{pmatrix}\}
```

The function PauliDecompose returns the expansion coefficients in the Pauli basis.

```
In[4]:= cc = PauliDecompose[L];
MatrixForm/@cc
Out[4]= \{2, \frac{i}{2}, -\frac{3}{2}, -1\}
```

Indeed, the coefficients reconstructs the original matrix.

```
In[5]:= new = cc.bs;
new // MatrixForm
Out[5]//MatrixForm=

$$\begin{pmatrix} 1 & 2i \\ -i & 3 \end{pmatrix}$$

In[6]:= L - new // Chop
Out[6]= \{\{0, 0\}, \{0, 0\}\}
```

Let us further consider a 2×2 Hermitian matrix.

```
In[7]:= H = RandomHermitian[];
H // MatrixForm
Out[7]//MatrixForm=

$$\begin{pmatrix} 0.558658 + 0.i & -0.89873 - 0.66087i \\ -0.89873 + 0.66087i & 0.120512 + 0.i \end{pmatrix}$$

In[8]:= H - Topple[H] // Chop // MatrixForm
Out[8]//MatrixForm=

$$\begin{pmatrix} 0 & 0 \\ 0 & 0 \end{pmatrix}$$

```

It is noted that the expansion coefficients are all real.

```
In[9]:= cc = PauliDecompose[H] // Chop
Out[9]= \{0.339585, -0.89873, 0.66087, 0.219073\}
```

Finally, consider a 2×2 unitary matrix.

```
In[=]:= U = RandomUnitary[];
U // MatrixForm
Out[=]/MatrixForm=

$$\begin{pmatrix} -0.35467 + 0.843542 i & -0.196382 - 0.352251 i \\ -0.116783 + 0.386016 i & 0.526328 + 0.748553 i \end{pmatrix}$$


In[=]:= Topple[U].U // Chop // MatrixForm
Out[=]/MatrixForm=

$$\begin{pmatrix} 1. & 0. \\ 0. & 1. \end{pmatrix}$$

```

One can see that the column vector of the expansion coefficients is normalized.

```
In[=]:= cc = PauliDecompose[U]
Out[=]= {0.085829 + 0.796047 i, -0.156582 + 0.0168825 i,
0.369134 - 0.0397996 i, -0.440499 + 0.0474942 i}

In[=]:= Conjugate[cc].cc // Chop
Out[=]= 1.
```

In the above demonstration, we have used the Pauli operators for *unlabelled* qubits. One could use the Pauli operators for qubits with labels. Let us consider a system of two qubits, which are denoted by the symbol S.

Let[Qubit, S]

Consider an arbitrary 2×2 matrix.

```
In[=]:= mat = RandomInteger[{-3, 3}, {2, 2}];
mat // MatrixForm
Out[=]/MatrixForm=

$$\begin{pmatrix} 1 & 0 \\ 1 & -1 \end{pmatrix}$$

```

This converts the matrix into an operator expression in terms of the Pauli operators on the labelled qubits. Here S[μ] corresponds to Pauli[μ] acting on the the qubit S[None].

```
In[=]:= op = Elaborate@ExpressionFor[mat, S[None]]

$$\frac{S^x}{2} - \frac{i S^y}{2} + S^z$$

Out[=]=
```

The operator expression can be converted back to a matrix by using Matrix.

```
In[=]:= new = Matrix[op];
new // MatrixForm
Out[=]/MatrixForm=

$$\begin{pmatrix} 1 & 0 \\ 1 & -1 \end{pmatrix}$$

```

Definition B.1 (*vector space of linear maps*) Let \mathcal{V} and \mathcal{W} be vector spaces. Let $\mathcal{L}(\mathcal{V}, \mathcal{W})$ be the set of all linear maps $\hat{L} : \mathcal{V} \rightarrow \mathcal{W}$. Equip it with a natural multiplication of linear map \hat{L} by scalars $x \in \mathbb{C}$ as

$$(x\hat{L})|v\rangle := x(\hat{L}|v\rangle) \quad (\text{B.1})$$

for all $|v\rangle \in \mathcal{V}$. Also define the sum of two linear maps by

$$(\hat{L} + \hat{M})|v\rangle := \hat{L}|v\rangle + \hat{M}|v\rangle \quad (\text{B.2})$$

for all $|v\rangle \in \mathcal{V}$. Then the set $\mathcal{L}(\mathcal{V}, \mathcal{W})$ forms a vector space. When $\mathcal{V} = \mathcal{W}$, $\mathcal{L}(\mathcal{V}) \equiv \mathcal{L}(\mathcal{V}, \mathcal{V})$ is the vector space of all linear *operators* on \mathcal{V} .

Let $\{|v_1\rangle, \dots, |v_m\rangle\}$ and $\{|w_1\rangle, \dots, |w_n\rangle\}$ be bases of \mathcal{V} and \mathcal{W} , respectively. A natural choice for basis of $\mathcal{L}(\mathcal{V}, \mathcal{W})$ is

$$\{|w_k\rangle \langle v_j| : j = 1, \dots, m; k = 1, \dots, n\}. \quad (\text{B.3})$$

$\mathcal{L}(\mathcal{V}, \mathcal{W})$ also needs a Hermitian product. In the same spirit as the Frobenius inner product of matrices, a natural choice of the Hermitian product in $\mathcal{L}(\mathcal{V}, \mathcal{W})$ inherited from the Hermitian products in \mathcal{V} and \mathcal{W} is that

$$\langle \hat{L}, \hat{M} \rangle := \text{Tr } \hat{L}^\dagger \hat{M} = \sum_j \langle v_j, \hat{L}^\dagger \hat{M} v_j \rangle_{\mathcal{V}} = \sum_j \langle \hat{L} v_j, \hat{M} v_j \rangle_{\mathcal{W}}. \quad (\text{B.4})$$

It is called the *trace Hermitian product* or simply *trace product*. With respect to this Hermitian product, the basis in Eq. (B.3) is orthonormal.

For the vector space $\mathcal{L}(\mathcal{V})$ of all *operators* on \mathcal{V} , equipped with the trace Hermitian product analogous to (B.4), another choice of basis other than the standard basis (B.3) is widely used. It is to pick the identity operator \hat{I} as an element of the basis. Then every other element in the basis must be *traceless*. For example, let \mathcal{S} be a two-dimensional Hilbert space, and in $\mathcal{L}(\mathcal{S})$ the four Pauli operators form such a basis, $\{\hat{I} \equiv \hat{S}^0, \hat{S}^x, \hat{S}^y, \hat{S}^z\}$. Obviously, the non-identity three Pauli operators are traceless, $\text{Tr } \hat{S}^\mu = 0$ ($\mu = x, y, z$). Any operator $\hat{A} \in \mathcal{L}(\mathcal{S})$ is expanded in the four Pauli operators

$$\hat{A} = \hat{S}^0 \alpha_0 + \hat{S}^x \alpha_x + \hat{S}^y \alpha_y + \hat{S}^z \alpha_z \quad (\alpha_\mu \in \mathbb{C}). \quad (\text{B.5})$$

The expansion coefficients can be obtained using the orthogonality of the basis and the trace Hermitian product,

$$\alpha_\mu = \frac{1}{2} \text{Tr } \hat{S}^\mu \hat{A} \quad (\text{B.6})$$

(recall that the Pauli operators are all Hermitian). We have already observed this fact in Exercise B.2 using the matrix form of the Pauli operators.

Example B.3 Consider the Hilbert space $\mathcal{S} \otimes \mathcal{S}$ associated with a system of two qubits. Show that the products of the Pauli operators

$$\{\hat{S}_1^\mu \otimes \hat{S}_2^\nu\} \quad (\text{B.7})$$

form an orthogonal basis of $\mathcal{L}(\mathcal{S} \otimes \mathcal{S})$.

Solution: Consider an arbitrary 4×4 matrix.



```
In[7]:= mat = RandomInteger[{-3, 3}, {4, 4}];
mat // MatrixForm
Out[7]//MatrixForm=
```

$$\begin{pmatrix} -2 & -1 & -1 & 1 \\ 1 & 0 & 1 & -1 \\ 2 & 2 & 0 & 2 \\ -2 & -3 & 0 & -3 \end{pmatrix}$$

This converts the matrix into an operator expression in terms of the products of the Pauli operators, which are represented by `Pauli`[μ, ν, \dots].

```
In[8]:= op = Elaborate@ExpressionFor[mat]
Out[8]= -\frac{5}{4} \sigma^0 \otimes \sigma^0 + \frac{\sigma^0 \otimes \sigma^x}{2} + \frac{\sigma^0 \otimes \sigma^z}{4} - \frac{3 \sigma^x \otimes \sigma^0}{4} + \frac{\sigma^x \otimes \sigma^x}{2} + i \sigma^x \otimes \sigma^y + \frac{5 \sigma^x \otimes \sigma^z}{4} - \frac{1}{4} i \sigma^y \otimes \sigma^0 +
\frac{1}{2} i \sigma^y \otimes \sigma^x + \sigma^y \otimes \sigma^y - \frac{5}{4} i \sigma^y \otimes \sigma^z + \frac{\sigma^z \otimes \sigma^0}{4} - \frac{\sigma^z \otimes \sigma^x}{2} - i \sigma^z \otimes \sigma^y - \frac{5 \sigma^z \otimes \sigma^z}{4}
```

This converts the operator expression back into the original matrix.

```
In[9]:= new = Matrix[op];
new // MatrixForm
Out[9]//MatrixForm=
```

$$\begin{pmatrix} -2 & -1 & -1 & 1 \\ 1 & 0 & 1 & -1 \\ 2 & 2 & 0 & 2 \\ -2 & -3 & 0 & -3 \end{pmatrix}$$

In the above demonstration, we have used the Pauli operators for *unlabelled* qubits. One could use the Pauli operators for qubits with labels. Let us consider a system of two qubits, which are denoted by the symbol `S` and the flavor indices.

`Let[Qubit, S]`

This converts the matrix into an operator expression in terms of the Pauli operators on the labelled qubits. Here `S[1, μ]` corresponds to `Pauli[$\mu, 0$]` acting on the first qubit and `S[2, μ]` to `Pauli[$0, \mu$]` on the second qubit.

```
In[10]:= op = ExpressionFor[mat, S[{1, 2}], None];
Elaborate[op]
Out[10]= -\frac{5}{4} - \frac{5}{4} S_1^z S_2^z - \frac{3}{2} S_1^z S_2^+ + \frac{1}{2} S_1^z S_2^- + S_1^+ S_2^+ + S_1^+ S_2^- +
\frac{5}{2} S_1^- S_2^z + 2 S_1^- S_2^+ - 2 S_1^- S_2^- + \frac{S_1^z}{4} - S_1^+ - \frac{S_1^-}{2} + \frac{S_2^z}{4} + \frac{S_2^+}{2} + \frac{S_2^-}{2}
-\frac{5}{4} + \frac{1}{2} S_1^x S_2^x + i S_1^x S_2^y + \frac{5}{4} S_1^x S_2^z + \frac{1}{2} i S_1^y S_2^x + S_1^y S_2^y -
\frac{5}{4} i S_1^y S_2^z - \frac{1}{2} S_1^z S_2^x - i S_1^z S_2^y - \frac{5}{4} S_1^z S_2^z - \frac{3 S_1^x}{4} - \frac{i S_1^y}{4} + \frac{S_1^z}{4} + \frac{S_2^x}{2} + \frac{S_2^z}{4}
```

The operator expression can be converted back to a matrix by using `Matrix`.

```
In[11]:= new = Matrix[op];
new // MatrixForm
Out[11]//MatrixForm=
```

$$\begin{pmatrix} -2 & -1 & -1 & 1 \\ 1 & 0 & 1 & -1 \\ 2 & 2 & 0 & 2 \\ -2 & -3 & 0 & -3 \end{pmatrix}$$

B.2 Superoperators

As the operators on vector spaces are vectors themselves, one can consider a linear map $\mathcal{F} : \mathcal{L}(\mathcal{V}) \rightarrow \mathcal{L}(\mathcal{W})$ from operators on \mathcal{V} to those on \mathcal{W} . We call it a *super-mapping* or *supermap* to distinguish it from one between simple vectors. Physically, supermaps are most relevant when input operators represent mixed states, that is, when they are density operators.

In many cases, the input and output spaces are identical, $\mathcal{V} = \mathcal{W}$. In such a case, \mathcal{F} itself is an operator—an operator on operators—and is called a *superoperator* on \mathcal{V} . Superoperators are useful to mathematically describe the evolution of open quantum systems, i.e., the systems interacting with other surrounding systems.

B.2.1 Matrix Representation

How can a supermap be characterized? Recall that a linear map of simple vectors is characterized by its matrix representation. Upon a choice of bases $\{|v_j\rangle\}$ and $\{|w_i\rangle\}$ of \mathcal{V} and \mathcal{W} , respectively, $\hat{L} : \mathcal{V} \rightarrow \mathcal{W}$ is completely specified by

$$\hat{L}|v_j\rangle = \sum_i |w_i\rangle L_{ij}. \quad (\text{B.8})$$

For a supermap $\mathcal{F} : \mathcal{L}(\mathcal{V}) \rightarrow \mathcal{L}(\mathcal{W})$, the involved spaces $\mathcal{L}(\mathcal{V})$ and $\mathcal{L}(\mathcal{W})$ have additional algebraic structures, and there are several ways to characterize it at different levels. One straightforward way to characterize a supermap is to take a plain analogy of the above matrix representation. Recall that $|v_k\rangle\langle v_l|$ and $|w_i\rangle\langle w_j|$ form the standard bases of $\mathcal{L}(\mathcal{V})$ and $\mathcal{L}(\mathcal{W})$, respectively. For each $|v_k\rangle\langle v_l|$, $\mathcal{F}(|v_k\rangle\langle v_l|)$ belongs to $\mathcal{L}(\mathcal{W})$ and is expanded in the standard basis $\{|w_i\rangle\langle w_j|\}$ [see Eq. (B.3)] as (notice the order of indices in $C_{ik;jl}$)

$$\mathcal{F}(|v_k\rangle\langle v_l|) = \sum_{ij} |w_i\rangle\langle w_j| C_{ik;jl}, \quad C_{ik;jl} \in \mathbb{C}. \quad (\text{B.9})$$

Here the matrix C —regarding (ik) and (jl) as collective indices—is called the *Choi matrix* associated with the supermap \mathcal{F} , and completely characterizes the supermap \mathcal{F} . For an arbitrary linear operator $\hat{\rho} := \sum_{kl} |v_k\rangle\langle v_l| \rho_{kl} \in \mathcal{L}(\mathcal{V})$, its image through \mathcal{F} is given by

$$\hat{\sigma} := \mathcal{F}(\hat{\rho}) = \sum_{kl} \mathcal{F}(|v_k\rangle\langle v_l|) \rho_{kl} = \sum_{ij} \sum_{kl} |w_i\rangle\langle w_j| C_{ik;jl} \rho_{kl}. \quad (\text{B.10})$$

This implies that the matrix elements (in the standard tensor-product basis) of $\hat{\sigma}$ and $\hat{\rho}$ are related to each other by the Choi matrix as

$$\sigma_{ij} = \sum_{kl} C_{ik;jl} \rho_{kl}. \quad (\text{B.11})$$

We consider supermaps transforming operators of the form

```
In[1]:= Let[Complex, ρ]
ρho = ρ@{0, 1, 2, 3}.S[Full]
Out[1]= S0 ρ0 + SX ρ1 + SY ρ2 + SZ ρ3
```

Here is a supermap, specified by a set of three Kraus elements.

```
In[2]:= ops = {2 S[4], S[5], S[6]};
spr = Supermap[ops]
Out[2]= Supermap[{2 S+, S-, SH}]
```

Under the supermap, the operator `rho` transforms as follows.

```
In[3]:= new = spr[rho] // Elaborate
Out[3]= -SY ρ2 +  $\frac{1}{2} \times (7 \rho_0 - 3 \rho_3) + S^Z \left( \frac{3 \rho_0}{2} + \rho_1 - \frac{5 \rho_3}{2} \right) + S^X \rho_3$ 
```

This gives the Choi matrix corresponding to the supermap.

```
In[4]:= tsr = ChoiMatrix[spr];
Dimensions@tsr
Out[4]= {2, 2, 2, 2}
```

The Choi matrix provides a matrix representation of the supermap. This is illustrated by the transformation of the elements of the matrix representation of the operator `rho`.

```
In[5]:= new2 = TensorContract[TensorProduct[tsr, Matrix@rho], {{2, 5}, {4, 6}}];
new2 // Simplify // MatrixForm
Out[5]/MatrixForm=

$$\begin{pmatrix} 5 \rho_0 + \rho_1 - 4 \rho_3 & i \rho_2 + \rho_3 \\ -i \rho_2 + \rho_3 & 2 \rho_0 - \rho_1 + \rho_3 \end{pmatrix}$$

In[6]:= new - ExpressionFor[new2, S] // Elaborate
Out[6]= 0
```

Let us take a few examples: First, consider a supermap of the simplest form

$$\mathcal{F}(\hat{\rho}) = \hat{A}\hat{\rho}\hat{B}^\dagger, \quad (\text{B.12})$$

where $\hat{A}, \hat{B} \in \mathcal{L}(\mathcal{V}, \mathcal{W})$. Then the Choi matrix of \mathcal{F} is given by

$$C_{ij;kl} = A_{ij}B_{kl}^*. \quad (\text{B.13})$$

Next, consider a supermap of the form

$$\mathcal{F}(\hat{\rho}) = -i[\hat{H}, \hat{\rho}]. \quad (\text{B.14})$$

It is the coherent part of the Lindblad equation—see Section 5.3. One can use the result in (B.13) putting either $\hat{A} = \hat{I}$ or $\hat{B} = \hat{I}$. It immediately follows that the Choi matrix for the supermap specified in (B.14)

$$C_{ij;kl} = -i(H_{ij}\delta_{kl} - \delta_{ij}H_{kl}^*). \quad (\text{B.15})$$

B.2.2 Operator-Sum Representation

Another method to characterize a supermap is the so-called operator-sum representation and turns out to be extremely useful in many areas of physics, including quantum information theory and quantum statistical mechanics. Putting the identity $\rho_{kl} = \langle v_k | \hat{\rho} | v_l \rangle$ back into (B.10), one gets

$$\mathcal{F}(\hat{\rho}) = \sum_{ij} \sum_{kl} |w_i\rangle \langle v_k | \hat{\rho} | v_l \rangle \langle w_j| C_{ik;jl}. \quad (\text{B.16})$$

Now identify $\hat{E}_{ik} := |w_i\rangle \langle v_k|$ as a linear map from \mathcal{V} to \mathcal{W} , $\hat{E}_{ik} \in \mathcal{L}(\mathcal{V}, \mathcal{W})$. Similarly, $|v_l\rangle \langle w_j| \in \mathcal{L}(\mathcal{W}, \mathcal{V})$ and it is identical to \hat{E}_{jl}^\dagger . Hence

$$\mathcal{F}(\hat{\rho}) = \sum_{ij} \sum_{kl} \hat{E}_{ik} \hat{\rho} \hat{E}_{jl}^\dagger C_{ik;jl}. \quad (\text{B.17}')$$

With the notation of collective indices $\mu \equiv (ik)$ and $\nu \equiv (jl)$, the supermap \mathcal{F} takes the operator-sum representation

$$\mathcal{F}(\hat{\rho}) = \sum_{\mu=1}^{MN} \sum_{\nu=1}^{MN} \hat{E}_\mu \hat{\rho} \hat{E}_\nu^\dagger C_{\mu\nu}, \quad (\text{B.17})$$

where $M := \dim \mathcal{V}$ and $N := \dim \mathcal{W}$. Diagrammatically, it is depicted as

$$\begin{array}{ccc} \mathcal{V} & \xleftarrow{\hat{E}_\nu^\dagger} & \mathcal{W} \\ \downarrow \hat{\rho} & & \downarrow \mathcal{E}(\hat{\rho}) \\ \mathcal{V} & \xrightarrow{\hat{E}_\mu} & \mathcal{W} \end{array} \quad (\text{B.18})$$

In Eqs. (B.17) and (B.17'), a standard basis $\{\hat{E}_\mu\}$ has been chosen in $\mathcal{L}(\mathcal{V}, \mathcal{W})$. But one can choose any basis, which leads to the following theorem.

Theorem B.2 Let \mathcal{V} and \mathcal{W} be vector spaces. If $\mathcal{F} : \mathcal{L}(\mathcal{V}) \rightarrow \mathcal{L}(\mathcal{W})$ is a supermap, then there exist $\hat{F}_\mu \in \mathcal{L}(\mathcal{V}, \mathcal{W})$ such that

$$\mathcal{F}(\hat{\rho}) = \sum_{\mu=1}^{MN} \sum_{\nu=1}^{MN} \hat{F}_\mu \hat{\rho} \hat{F}_\nu^\dagger C_{\mu\nu}, \quad C_{\mu\nu} \in \mathbb{C}. \quad (\text{B.19})$$

Consider a supermap specified by a set of operator and a matrix of coefficients.

```
In[7]:= ops = {I, S[1], S[2], S[3]}
Out[7]= {I, Sx, Sy, Sz}

In[8]:= Let[Complex, c]
cc = Array[c, {4, 4}];
cc // MatrixForm
Out[8]//MatrixForm=

```

$$\begin{pmatrix} c_{1,1} & c_{1,2} & c_{1,3} & c_{1,4} \\ c_{2,1} & c_{2,2} & c_{2,3} & c_{2,4} \\ c_{3,1} & c_{3,2} & c_{3,3} & c_{3,4} \\ c_{4,1} & c_{4,2} & c_{4,3} & c_{4,4} \end{pmatrix}$$

Here is the supermap. It represents a map from operators to other operators.

```
In[1]:= spr = Supermap[ops, cc]
Out[1]= Supermap[{i, Sx, Sy, Sz}, {{c1,1, c1,2, c1,3, c1,4}, {c2,1, c2,2, c2,3, c2,4}, {c3,1, c3,2, c3,3, c3,4}, {c4,1, c4,2, c4,3, c4,4}}]
In[2]:= Let[Complex, ρ]
ρho = ρ[{0, 1, 2, 3}].S[Full]
Out[2]= S0 ρ0 + Sx ρ1 + Sy ρ2 + Sz ρ3
```

This is the result acting the supermap on the above operator.

```
In[3]:= new = spr[rho]
Out[3]= C2,2 ρ0 + C3,3 ρ0 + C4,4 ρ0 + C1,1 S0 ρ0 + i C1,2 ρ1 - i C2,1 ρ1 - i C3,4 ρ1 + i C4,3 ρ1 +
i C1,3 ρ2 + i C2,4 ρ2 - i C3,1 ρ2 - i C4,2 ρ2 + i C1,4 ρ3 - i C2,3 ρ3 + i C3,2 ρ3 - i C4,1 ρ3 +
Sx (i C1,2 ρ0 - i C2,1 ρ0 + i C3,4 ρ0 - i C4,3 ρ0 + C1,1 ρ1 + C2,2 ρ1 - C3,3 ρ1 - C4,4 ρ1 -
C1,4 ρ2 + C2,3 ρ2 + C3,2 ρ2 - C4,1 ρ2 + C1,3 ρ3 + C2,4 ρ3 + C3,1 ρ3 + C4,2 ρ3) +
Sy (i C1,3 ρ0 - i C2,4 ρ0 - i C3,1 ρ0 + i C4,2 ρ0 + C1,4 ρ1 + C2,3 ρ1 + C3,2 ρ1 + C4,1 ρ1 +
C1,1 ρ2 - C2,2 ρ2 + C3,3 ρ2 - C4,4 ρ2 - C1,2 ρ3 - C2,1 ρ3 + C3,4 ρ3 + C4,3 ρ3) +
Sz (i C1,4 ρ0 + i C2,3 ρ0 - i C3,2 ρ0 - i C4,1 ρ0 - C1,3 ρ1 + C2,4 ρ1 - C3,1 ρ1 + C4,2 ρ1 +
C1,2 ρ2 + C2,1 ρ2 + C3,4 ρ2 + C4,3 ρ2 + C1,1 ρ3 - C2,2 ρ3 - C3,3 ρ3 + C4,4 ρ3)
```

We are often interested in mapping density operators—not just any operators. In this case, the relevant supermaps are required to preserve the properties of density operators—density operators are *Hermitian* and in particular *positive*. The condition to preserve Hermiticity simplifies the representation (B.17) further.

Theorem B.3 Let \mathcal{V} and \mathcal{W} be vector spaces, equipped with Hermitian products.

Let $\mathcal{F} : \mathcal{L}(\mathcal{V}) \rightarrow \mathcal{L}(\mathcal{W})$ be a supermap. If $\mathcal{F}(\hat{\rho})$ is Hermitian for every Hermitian $\hat{\rho} \in \mathcal{L}(\mathcal{V})$, then there exist $\hat{F}_\mu \in \mathcal{L}(\mathcal{V}, \mathcal{W})$ such that

$$\mathcal{F}(\hat{\rho}) = \sum_{\mu=1}^{MN} \epsilon_\mu \hat{F}_\mu \hat{\rho} \hat{F}_\mu^\dagger, \quad (\text{B.20})$$

where $\epsilon_\mu = \pm 1$ and the linear maps \hat{F}_μ are orthogonal to each other, $\text{Tr } \hat{F}_\mu^\dagger \hat{F}_\nu = 0$ for $\mu \neq \nu$.

Consider a supermap specified by a set of operator and a vector of coefficients.

```
In[1]:= ops = {I, S[1], S[2], S[3]}
Out[1]= {i, Sx, Sy, Sz}
In[2]:= Let[Real, c]
cc = Array[c, 4];
cc // MatrixForm
Out[2]//MatrixForm=

$$\begin{pmatrix} c_1 \\ c_2 \\ c_3 \\ c_4 \end{pmatrix}$$

```

Here is the supermap. It represents a map from operators to other operators.

```
In[=]:= spr = Supermap[ops, cc]
Out[=]:= Supermap[{i, Sx, Sy, Sz}, {c1, c2, c3, c4}]

In[=]:= Let[Complex, ρ]
ρho = ρ[{0, 1, 2, 3}].S[Full]
Out[=]:= S0 ρ0 + Sx ρ1 + Sy ρ2 + Sz ρ3
```

This is the result acting the supermap on the above operator.

```
In[=]:= new = spr[rho]
Out[=]:= (c2 + c3 + c4) ρ0 + c1 S0 ρ0 + (c1 + c2 - c3 - c4) Sx ρ1 +
(c1 - c2 + c3 - c4) Sy ρ2 + (c1 - c2 - c3 + c4) Sz ρ3
```

In the representation (B.20), all numerical factors have been absorbed into the operators \hat{F}_μ leaving only possibly negative signs in ϵ_μ . An immediate question is what condition a supermap should satisfy to have $\epsilon_\mu = 1$ for all μ ? Would the condition to preserve positivity be sufficient to guarantee it? Unfortunately, the positivity-preserving condition does not bring any meaningful simplification, and a much stronger condition is required:

Definition B.4 (*completely positive supermap*) Let \mathcal{V} and \mathcal{W} be vector spaces and $\mathcal{F} : \mathcal{L}(\mathcal{V}) \rightarrow \mathcal{L}(\mathcal{W})$ a supermap. \mathcal{F} is said to be *completely positive* if $\mathcal{F} \otimes \mathcal{I} : \mathcal{L}(\mathcal{V} \otimes \mathcal{E}) \rightarrow \mathcal{L}(\mathcal{W} \otimes \mathcal{E})$ is positive^{B.1} for any vector space \mathcal{E} , where \mathcal{I} denotes the identity superoperator on $\mathcal{L}(\mathcal{E})$.

Physically, the vector space \mathcal{E} is associated with an environment—see Section 5.1. The operator-sum representation in (B.19) or (B.20) is further simplified for completely positive supermaps. The following example exhibits the motivation.

Exercise B.4 For any linear maps $\hat{F}_\mu \in \mathcal{L}(\mathcal{V}, \mathcal{W})$, the supermap $\mathcal{F} : \mathcal{L}(\mathcal{V}) \rightarrow \mathcal{L}(\mathcal{W})$ defined by

$$\mathcal{F}(\hat{\rho}) := \sum_{\mu} \hat{F}_{\mu} \hat{\rho} \hat{F}_{\mu}^{\dagger} \quad (\text{B.21})$$

is completely positive.

Note that the linear maps \hat{F}_μ in (B.21) are completely arbitrary. They do not have to be orthogonal to each other, $\text{Tr } \hat{F}_{\mu}^{\dagger} \hat{F}_{\nu} \neq 0$, nor to span the space $\mathcal{L}(\mathcal{V}, \mathcal{W})$. The following theorem confirms that any supermap takes the above form in Eq. (B.21). In fact, for a given supermap, one can find a more compact and refined linear maps to represent it with.

Theorem B.5 (*Kraus representation theorem*) Let \mathcal{V} and \mathcal{W} be vector spaces, and $\mathcal{F} : \mathcal{L}(\mathcal{V}) \rightarrow \mathcal{L}(\mathcal{W})$ be a supermap. Then the following statement are equivalent:

- (a) \mathcal{F} is completely positive.

^{B.1}Here “positive” actually means “positivity-preserving”.

(b) For any $\hat{\rho} \in \mathcal{L}(\mathcal{V})$, the effect $\mathcal{F}(\hat{\rho})$ can be written as

$$\mathcal{F}(\hat{\rho}) = \sum_{\mu=0}^{m-1} \hat{F}_\mu \hat{\rho} \hat{F}_\mu^\dagger, \quad (\text{B.22})$$

where $m \leq (\dim \mathcal{V})(\dim \mathcal{W})$ and $\hat{F}_\mu : \mathcal{V} \rightarrow \mathcal{W}$ are *mutually orthogonal* linear maps— $\text{Tr } \hat{F}_\mu^\dagger \hat{F}_\nu = 0$ for all $\mu \neq \nu$.

(c) For any $\hat{\rho} \in \mathcal{L}(\mathcal{V})$, the effect $\mathcal{F}(\hat{\rho})$ can be written as a finite sum of the form

$$\mathcal{F}(\hat{\rho}) = \sum_{\mu} \hat{F}_\mu \hat{\rho} \hat{F}_\mu^\dagger, \quad (\text{B.23})$$

where $\hat{F}_\mu : \mathcal{V} \rightarrow \mathcal{W}$ are (arbitrary) linear maps.

The expressions (B.22) and (B.23) are called the *Kraus operator-sum representation* or simply the *Kraus representation* of the completely positive supermap \mathcal{F} . The linear maps \hat{F}_μ are called the *Kraus elements* or the *Kraus maps* (the *Kraus operators* when $\mathcal{V} = \mathcal{W}$). The *orthogonal* Kraus elements in Eq. (B.22) are optimal in the sense that the sum has the least possible number of terms.

A completely positive supermap can be specified by a set of Kraus elements.

```
In[1]:= ops = { S[4], S[5], S[3] }
Out[1]= {S^+, S^-, S^z}
```

Here is the supermap. It represents a map from operators to other operators.

```
In[2]:= spr = Supermap[ops]
Out[2]= Supermap[{S^+, S^-, S^z}]

In[3]:= Let[Complex, rho]
rho = rho[{0, 1, 2, 3}].S[Full]
Out[3]= S^0 rho_0 + S^x rho_1 + S^y rho_2 + S^z rho_3
```

This is the result acting the supermap on the above operator.

```
In[4]:= new = spr[rho]
Out[4]= 2 rho_0 - S^x rho_1 - S^y rho_2
```

It is fairly obvious that a supermap expressed in the form (B.22) or (B.23) is completely positive. One can prove the converse starting from (B.17').

Exercise B.5 Using the representation in (B.17') and requiring the positivity of $(\mathcal{F} \otimes \mathcal{I})(|\Phi\rangle\langle\Phi|)$ with

$$|\Phi\rangle := \sum_j |v_j\rangle \otimes |v_j\rangle \in \mathcal{V} \otimes \mathcal{V}, \quad (\text{B.24})$$

prove that a completely positive map has the Kraus representation of the form (B.22).

B.2.3 Choi Isomorphism

A less widely known yet intriguing method to characterize a supermap is provided by the *Choi isomorphism* (also known as Jamiolkowski, Choi-Jamiolkowski or Jamiolkowski-Choi isomorphism).^{B.2}

Before we discuss the Choi isomorphism of supermaps, let us first examine the same isomorphism of linear maps. Let $\hat{A} : \mathcal{V} \rightarrow \mathcal{W}$ be a linear map. Recall that it is completely characterized by the $n \times m$ matrix A such that

$$\hat{A} = \sum_{kj} |w_k\rangle A_{kj} \langle v_j|, \quad (\text{B.25})$$

where $\{v_j\}$ and $|w_k\rangle$ are bases of \mathcal{V} and \mathcal{W} , respectively. Now note that the same matrix defines a vector

$$|A\rangle := \sum_{kj} |w_k\rangle \otimes |v_j\rangle A_{kj} \quad (\text{B.26})$$

in the tensor-product space $\mathcal{W} \otimes \mathcal{V}$. The correspondence $\hat{A} \leftrightarrow |A\rangle$ turns out to be an isomorphism between $\mathcal{L}(\mathcal{V}, \mathcal{W})$ and $\mathcal{W} \otimes \mathcal{V}$. The isomorphism is called the Choi isomorphism and $|A\rangle$ is called the *Choi vector* associated with the linear map \hat{A} . Looking almost trivial at a first glance, the isomorphism brings about several interesting things. To see it, consider a maximally entangled state

$$|\Phi\rangle := \sum_k |v_k\rangle \otimes |v_k\rangle \in \mathcal{V} \otimes \mathcal{V}. \quad (\text{B.27})$$

in the tensor-product space $\mathcal{V} \otimes \mathcal{V}$. First, observe that the Choi vector $|A\rangle$ of a linear map \hat{A} is given by

$$|A\rangle = (\hat{A} \otimes \hat{I}) |\Phi\rangle. \quad (\text{B.28})$$

This is depicted in the following quantum circuit model

$$|A\rangle = |\Phi\rangle \left\{ \begin{array}{c} \xrightarrow{\quad} \\ \boxed{\hat{A}} \\ \xrightarrow{\quad} \end{array} \right. \quad (\text{B.29})$$

The isomorphism preserves the Hermitian products in $\mathcal{L}(\mathcal{V}, \mathcal{W})$ and $\mathcal{W} \otimes \mathcal{V}$, that is, $\text{Tr } \hat{A} \hat{B} = \langle A | B \rangle$ for all linear maps \hat{A} and \hat{B} . Further, for an arbitrary state $|\psi\rangle = \sum_j |v_j\rangle \psi_j \in \mathcal{V}$, define its conjugate state by

$$|\psi^*\rangle := \sum_j |v_j\rangle \psi_j^*. \quad (\text{B.30})$$

Then, it follows that

$$\hat{A} |\psi\rangle = \langle \psi^* | (\hat{A} \otimes \hat{I}) |\Phi\rangle = \langle \psi^* | A \rangle, \quad (\text{B.31})$$

^{B.2}There are subtle but important differences between the Choi and Jamiolkowski isomorphism; see [Jiang et al. \(2013\)](#).

where the Hermitian product on the right-hand side is applied partially and only on \mathcal{V} —the remaining part is a vector belonging to \mathcal{W} . This is depicted in a quantum circuit model as

$$|\Phi\rangle \left\{ \begin{array}{c} \text{---} \boxed{\hat{A}} \text{---} \hat{A} |\psi\rangle \\ \text{---} \bigtriangleup \text{---} |\psi^*\rangle \end{array} \right., \quad (\text{B.32})$$

where the quantum circuit element \bigtriangleup represents the projection onto the state specified at the output port. Interestingly, the result is not affected whether the projection onto $|\psi^*\rangle$ is made before or after the operation \hat{A} . This does not violate any physical principle as the two parts in $\mathcal{V} \otimes \mathcal{V}$ are separated spacelike.

Now let us turn to the Choi isomorphism between supermaps and operators: Consider again the maximally entangled state in Eq. (B.27) and operate an extended supermap $\mathcal{F} \otimes \mathcal{I} : \mathcal{L}(\mathcal{V} \otimes \mathcal{V}) \rightarrow \mathcal{L}(\mathcal{W} \otimes \mathcal{V})$ on $|\Phi\rangle \langle \Phi|$ to get

$$\begin{aligned} \hat{C}_{\mathcal{F}} := (\mathcal{F} \otimes \mathcal{I})(|\Phi\rangle \langle \Phi|) &= \sum_{kl} \mathcal{F}(|v_k\rangle \langle v_l|) \otimes |v_k\rangle \langle v_l| \\ &= \sum_{ij} \sum_{kl} |w_i v_k\rangle \langle w_j v_l| C_{ik;jl}, \end{aligned} \quad (\text{B.33})$$

where C is the Choi matrix of \mathcal{F} ; see Eq. (B.9). This is depicted in a quantum circuit model as

$$\hat{C}_{\mathcal{F}} = |\Phi\rangle \left\{ \begin{array}{c} \text{---} \boxed{\mathcal{F}} \text{---} \\ \text{---} \end{array} \right. \quad (\text{B.34})$$

Clearly $\hat{C}_{\mathcal{F}}$ is an operator (not a superoperator) on $\mathcal{W} \otimes \mathcal{V}$ and the Choi matrix C is nothing but its matrix representation in the standard tensor-product basis. Hence $\hat{C}_{\mathcal{F}}$ is called the *Choi operator* associated with \mathcal{F} . It turns out that the correspondence $\mathcal{F} \leftrightarrow \hat{C}_{\mathcal{F}}$ by means of (B.33) is one-to-one and an isomorphism.^{B.3} For any state $|\psi\rangle \in \mathcal{V}$, the effect $\mathcal{F}(|\psi\rangle \langle \psi|)$ of supermap \mathcal{F} on the pure state can be obtained by means of the conjugate state $|\psi^*\rangle$ [see Eq. (B.30)] as

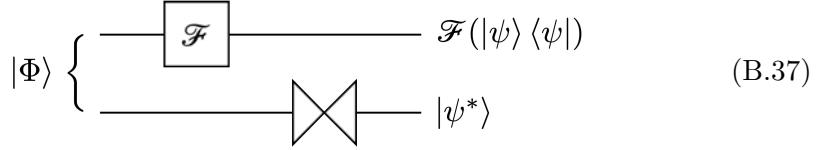
$$\mathcal{F}(|\psi\rangle \langle \psi|) = \langle \psi^* | \hat{C}_{\mathcal{F}} | \psi^* \rangle, \quad (\text{B.35})$$

or, more generally, for any $\hat{\rho} = \sum_{ij} |v_i\rangle \langle v_j| \rho_{ij}$

$$\mathcal{F}(\hat{\rho}) = \text{Tr}_{\mathcal{V}} \hat{\rho}^* \hat{C}_{\mathcal{F}}, \quad (\text{B.36})$$

^{B.3}Here we have just defined an association $\mathcal{F} \mapsto \hat{C}_{\mathcal{F}}$. Given an operator $\hat{C} \in \mathcal{L}(\mathcal{W} \otimes \mathcal{V})$, one can also find the corresponding supermap $\mathcal{F}_{\hat{C}}$, that is, the association $\hat{C} \mapsto \mathcal{F}_{\hat{C}}$ in the reverse direction; see Størmer (2013).

where $\hat{\rho}^* := \sum_{ij} |v_i\rangle\langle v_j| \rho_{ij}^*$. This has been depicted in a quantum circuit model



Further, taking the matrix representation of each entity in the relation (B.36) confirms the linear transformation rule (B.11) between the matrix elements of $\hat{\rho}$ and $\hat{\sigma} := \mathcal{F}(\hat{\rho})$. The transformation rule in (B.11) is another illustration of the Choi isomorphism.

The Choi operator plays a key role in the *gate teleportation* protocol, and it provides an interesting proof of the Kraus-representation theorem (see Section 5.1).

B.3 Partial Trace

Tensor product (see Appendix A.5) extends vectors and operators. Partial trace is effectively an inverse procedure and reduces operators on a tensor-product space to one of component spaces. Consider an operator \hat{C} on a tensor product space $\mathcal{V} \otimes \mathcal{W}$. The partial trace over the space \mathcal{W} is defined by

$$\hat{A} = \text{Tr}_{\mathcal{W}} \hat{C} = \sum_j \langle w_j | \hat{C} | w_j \rangle \quad (\text{B.38})$$

The procedure is said to *trace out* the vector space \mathcal{W} , and the resulting operator \hat{A} is called a *reduced operator* of \hat{C} . Given the matrix representation $C_{ij;kl}$ of \hat{C} in the standard product basis

$$\hat{C} |v_k w_l\rangle = \sum_{ij} |v_i w_j\rangle C_{ij;kl}, \quad (\text{B.39})$$

one can obtain the matrix representation of the reduced operator \hat{A} in the basis $\{|v_i\rangle\}$ by

$$A_{ik} = \sum_j C_{ij;kj}. \quad (\text{B.40})$$

Consider an operator A on a three-qubit Hilbert space. Suppose that an 8x matrix A is its matrix representation.

```
In[7]:= A = Re@RandomMatrix[8];
A[[;; 5, ;; 5]] // MatrixForm
Out[7]//MatrixForm=
{{0.748769, 0.633463, 0.637068, 0.831058, 0.0608231},
 {-0.15509, -0.511914, -0.768065, -0.994727, 0.973921},
 {-0.772355, 0.33461, -0.861444, 0.0559188, 0.0583494},
 {-0.235148, -0.959995, 0.673298, -0.728021, -0.65737},
 {-0.605041, 0.584033, 0.906122, -0.787569, -0.509827}}
```

This is the reduced matrix after tracing out the second and third qubits.

```
In[=]:= redA1 = PartialTrace[A, {2, 3}];
redA1 // MatrixForm
Out[=]//MatrixForm=

$$\begin{pmatrix} -1.35261 & -2.02621 \\ -0.528031 & 0.439414 \end{pmatrix}$$

```

This traces out the second qubit.

```
In[=]:= redA2 = PartialTrace[A, {2}];
redA2 // MatrixForm
Out[=]//MatrixForm=

$$\begin{pmatrix} -0.446217 & -0.0737435 & -0.0971918 & 1.01527 \\ -1.01302 & 0.155287 & -0.224515 & -1.16846 \\ 0.331291 & 0.216007 & 0.112729 & 1.26334 \\ 0.736839 & 0.0580864 & -0.329636 & -0.0437415 \end{pmatrix}$$

```

Example B.6 Show that partial trace is a *completely positive supermap* (see Definition B.4).

One can prove it simply by constructing an operator-sum representation as in Theorem B.5. Consider the partial trace over the subspace \mathcal{W} . According to the definition of the partial trace, we write

$$\text{Tr}_{\mathcal{W}} \hat{C} = \sum_{ijk} |v_i\rangle \langle v_i w_j| \hat{C} |v_k w_j\rangle \langle v_k| \quad (\text{B.41})$$

It means that defining $\hat{F}_j = \sum_i |v_i\rangle \langle v_i w_j|$ the partial trace can be expressed as

$$\text{Tr}_{\mathcal{W}} \hat{C} = \sum_j \hat{F}_j \hat{C} \hat{F}_j^\dagger, \quad (\text{B.42})$$

which proves the claim.

B.4 Partial Transposition

We conclude this appendix with a rather unusual mathematical tool—the *partial transposition*. As the name suggests, it applies the matrix transposition to the part of the matrix representation of an operator which corresponds to a subsystem. The resulting matrix gives a new operator associated with it. Roughly speaking, the partial transformation would correspond to a time-reversal transformation only on the subsystem.^{B.4}

Like the (full) transposition, the partial transposition *cannot* be a *linear supermap*. Nevertheless, the partial transposition has attracted considerable attention thanks to the seminal work on the separability test of mixed states of a composite quantum system by Peres (1996) and Horodecki *et al.* (1996). Since

^{B.4}Rigorously speaking, this statement is wrong because no anti-unitary transformation such as time reversal can be applied on a subpart of the system.

then, it has been widely used for the study of the structure of the tensor-product space of composite systems with respect to various entanglement properties.

Consider an operator \hat{A} on a tensor product space $\mathcal{V} \otimes \mathcal{W}$ with the matrix representation

$$\hat{A} = \sum_{ij;kl} |v_i w_j\rangle \langle v_k w_l| A_{ij;kl} \quad (\text{B.43})$$

in the standard tensor-product basis, $\{|v_i w_j\rangle \equiv |v_i\rangle \otimes |w_j\rangle\}$. The *partial transpose* of $\hat{A}^{\mathcal{W}}$ with respect to the subspace \mathcal{W} is defined by

$$\hat{A}^{\mathcal{W}} := \sum_{ij;kl} |v_i w_j\rangle \langle v_k w_l| A_{il;kj}. \quad (\text{B.44})$$

In a fixed basis, the partial transposition is entirely defined by the matrix representations. For the above case,

$$A_{ij;kl}^{\mathcal{W}} = A_{il;kj}. \quad (\text{B.45})$$

It is important to remember that the partial transposition is basis-dependent.

Consider a system consisting of two subsystem with Hilbert-space dimensions 2 and 3, respectively. A and B are matrix representations of operators on two respective systems.

```
In[=] := Let[Species, a, b]
          A = Array[a, {2, 2}];
          A // MatrixForm
          B = Array[b, {3, 3}];
          B // MatrixForm
Out[=] //MatrixForm=

$$\begin{pmatrix} a_{1,1} & a_{1,2} \\ a_{2,1} & a_{2,2} \end{pmatrix}$$

Out[=] //MatrixForm=

$$\begin{pmatrix} b_{1,1} & b_{1,2} & b_{1,3} \\ b_{2,1} & b_{2,2} & b_{2,3} \\ b_{3,1} & b_{3,2} & b_{3,3} \end{pmatrix}$$

```

Let us take a special case -- an operator with a single term of tensor product.

```
In[=] := matAB = CircleTimes[A, B];
          matAB // MatrixForm
Out[=] //MatrixForm=

$$\begin{pmatrix} a_{1,1} b_{1,1} & a_{1,1} b_{1,2} & a_{1,1} b_{1,3} & a_{1,2} b_{1,1} & a_{1,2} b_{1,2} & a_{1,2} b_{1,3} \\ a_{1,1} b_{2,1} & a_{1,1} b_{2,2} & a_{1,1} b_{2,3} & a_{1,2} b_{2,1} & a_{1,2} b_{2,2} & a_{1,2} b_{2,3} \\ a_{1,1} b_{3,1} & a_{1,1} b_{3,2} & a_{1,1} b_{3,3} & a_{1,2} b_{3,1} & a_{1,2} b_{3,2} & a_{1,2} b_{3,3} \\ a_{2,1} b_{1,1} & a_{2,1} b_{1,2} & a_{2,1} b_{1,3} & a_{2,2} b_{1,1} & a_{2,2} b_{1,2} & a_{2,2} b_{1,3} \\ a_{2,1} b_{2,1} & a_{2,1} b_{2,2} & a_{2,1} b_{2,3} & a_{2,2} b_{2,1} & a_{2,2} b_{2,2} & a_{2,2} b_{2,3} \\ a_{2,1} b_{3,1} & a_{2,1} b_{3,2} & a_{2,1} b_{3,3} & a_{2,2} b_{3,1} & a_{2,2} b_{3,2} & a_{2,2} b_{3,3} \end{pmatrix}$$

```

This is the partial trace of matAB with respect to the second subsystem.

```
In[6]:= new = PartialTranspose[matAB, {2, 3}, {2}];
new // MatrixForm
Out[6]//MatrixForm=
```

$$\begin{pmatrix} a_{1,1} b_{1,1} & a_{1,1} b_{2,1} & a_{1,1} b_{3,1} & a_{1,2} b_{1,1} & a_{1,2} b_{2,1} & a_{1,2} b_{3,1} \\ a_{1,1} b_{1,2} & a_{1,1} b_{2,2} & a_{1,1} b_{3,2} & a_{1,2} b_{1,2} & a_{1,2} b_{2,2} & a_{1,2} b_{3,2} \\ a_{1,1} b_{1,3} & a_{1,1} b_{2,3} & a_{1,1} b_{3,3} & a_{1,2} b_{1,3} & a_{1,2} b_{2,3} & a_{1,2} b_{3,3} \\ a_{2,1} b_{1,1} & a_{2,1} b_{2,1} & a_{2,1} b_{3,1} & a_{2,2} b_{1,1} & a_{2,2} b_{2,1} & a_{2,2} b_{3,1} \\ a_{2,1} b_{1,2} & a_{2,1} b_{2,2} & a_{2,1} b_{3,2} & a_{2,2} b_{1,2} & a_{2,2} b_{2,2} & a_{2,2} b_{3,2} \\ a_{2,1} b_{1,3} & a_{2,1} b_{2,3} & a_{2,1} b_{3,3} & a_{2,2} b_{1,3} & a_{2,2} b_{2,3} & a_{2,2} b_{3,3} \end{pmatrix}$$

Take close look at individual blocks.

```
In[7]:= new[[1 ;; 3, 1 ;; 3]] // MatrixForm
new[[1 ;; 3, 1 + 3 ;; 3 + 3]] // MatrixForm
Out[7]//MatrixForm=
```

$$\begin{pmatrix} a_{1,1} b_{1,1} & a_{1,1} b_{2,1} & a_{1,1} b_{3,1} \\ a_{1,1} b_{1,2} & a_{1,1} b_{2,2} & a_{1,1} b_{3,2} \\ a_{1,1} b_{1,3} & a_{1,1} b_{2,3} & a_{1,1} b_{3,3} \end{pmatrix}$$

$$\begin{pmatrix} a_{1,2} b_{1,1} & a_{1,2} b_{2,1} & a_{1,2} b_{3,1} \\ a_{1,2} b_{1,2} & a_{1,2} b_{2,2} & a_{1,2} b_{3,2} \\ a_{1,2} b_{1,3} & a_{1,2} b_{2,3} & a_{1,2} b_{3,3} \end{pmatrix}$$

This is the partial trace with respect to the first subsystem.

```
In[8]:= new = PartialTranspose[matAB, {2, 3}, {1}];
new // MatrixForm
Out[8]//MatrixForm=
```

$$\begin{pmatrix} a_{1,1} b_{1,1} & a_{1,1} b_{1,2} & a_{1,1} b_{1,3} & a_{2,1} b_{1,1} & a_{2,1} b_{1,2} & a_{2,1} b_{1,3} \\ a_{1,1} b_{2,1} & a_{1,1} b_{2,2} & a_{1,1} b_{2,3} & a_{2,1} b_{2,1} & a_{2,1} b_{2,2} & a_{2,1} b_{2,3} \\ a_{1,1} b_{3,1} & a_{1,1} b_{3,2} & a_{1,1} b_{3,3} & a_{2,1} b_{3,1} & a_{2,1} b_{3,2} & a_{2,1} b_{3,3} \\ a_{1,2} b_{1,1} & a_{1,2} b_{1,2} & a_{1,2} b_{1,3} & a_{2,2} b_{1,1} & a_{2,2} b_{1,2} & a_{2,2} b_{1,3} \\ a_{1,2} b_{2,1} & a_{1,2} b_{2,2} & a_{1,2} b_{2,3} & a_{2,2} b_{2,1} & a_{2,2} b_{2,2} & a_{2,2} b_{2,3} \\ a_{1,2} b_{3,1} & a_{1,2} b_{3,2} & a_{1,2} b_{3,3} & a_{2,2} b_{3,1} & a_{2,2} b_{3,2} & a_{2,2} b_{3,3} \end{pmatrix}$$

Appendix C

Group Theory

• August 3, 2021 (v1.6)

Group theory is the study of algebraic structures called groups. It has emerged as an abstraction unifying ideas of number theory, geometry, and the theory of algebraic equations. It forms the core part of abstract algebra.

The notion of group suits well physical conception of symmetry, and group theory has provided valuable theoretical tools to exploit the symmetry of physical problems. Crystallography was the first in physics to use group theory extensively. With the advent of quantum mechanics, group theory has occupied a key position at the center stage of physical theories in various areas ranging from condensed matter physics to high-energy physics.

In quantum information and related areas, the use of group theory is not restricted to symmetry considerations. It facilitates and boosts investigations into the algebraic structures of multi-partite quantum states and quantum operations. For example, group theory lays out elegant and efficient algebraic tools to develop quantum algorithms, construct quantum error-correction codes, and analyze quantum cryptosystems.

In this appendix, we introduce briefly elementary concepts and theorems concerning groups. A great number of textbooks are available for more complete accounts, including [Cornwell \(1984\)](#)—or [Cornwell \(1997\)](#) if an abridged version is preferred—as well as [Wigner \(1959\)](#) for physicists.

C.1 The Concept

Mathematically, a group is a set of algebraic objects that must obey certain *axioms*. The development of the theory does not depend on the specific nature of the elements themselves, but in most physical applications these elements are transformations of one kind or another. The *multiplication* between objects is key to determine the structure of a group. Physically, it corresponds to the composition—successive application—of transformations.

Definition C.1 (*group*) A set \mathcal{G} of elements is called *group* if the following four *group axioms* are satisfied:

- (a) There exists an operation which associates with every pair of elements $G_1 \in \mathcal{G}$ and $G_2 \in \mathcal{G}$ another element $G_3 \in \mathcal{G}$. This operation is called *multiplication* and is written as $G_3 = G_1G_2$, G_3 being described as the “product of G_1 and G_2 .”
- (b) For any three elements $G_1, G_2, G_3 \in \mathcal{G}$,

$$(G_1G_2)G_3 = G_1(G_2G_3), \quad (\text{C.1})$$

i.e., the multiplication is *associative*.

- (c) There exists an *identity element* $e \in \mathcal{G}$ such that

$$GE = EG = G \quad (\text{C.2})$$

for every element $G \in \mathcal{G}$.

- (d) For each element $G \in \mathcal{G}$ there exists an *inverse element* $G^{-1} \in \mathcal{G}$ such that

$$GG^{-1} = G^{-1}G = E. \quad (\text{C.3})$$

The *order* of a group \mathcal{G} is defined to be the number of elements in \mathcal{G} , which may be finite, countably infinite, or even non-countably infinite. It is denoted by $|\mathcal{G}|$. A group with finite order is called a *finite group*.

One of the most frequently appearing examples of group in quantum information is *Pauli group*. The Pauli group on a single qubit consists of the Pauli operators I , X , Y , and Z . Explicitly, it is given by

$$\mathcal{P}(1) = \{\pm I, \pm iI, \pm X, \pm iX, \pm Y, \pm iY, \pm Z, \pm iZ\}. \quad (\text{C.4})$$

The additional phase factors ± 1 and $\pm i$ are included because a multiplication of two Pauli operators may result in another Pauli operator with one of such phase factors.

As an example, consider the Pauli group on a single qubit.

```
In[7]:= grp = PauliGroup[S[1]]
Out[7]= PauliGroup[{S1}]

In[8]:= elm = GroupElements@grp;
elm // PauliForm
Out[8]= {I, X, Y, Z, -I, -X, -Y, -Z, i I, i X, i Y, i Z, -i I, -i X, -i Y, -i Z}
```

This shows a part of the group multiplication table in terms of the index of the elements in the group.

tbl = GroupMultiplicationTable@grp;								
TableForm[tbl[[;; 8, ;; 8]], TableHeadings → Automatic]								
Out[=]/TableForm=	1	2	3	4	5	6	7	8
1	1	2	3	4	5	6	7	8
2	2	1	12	15	6	5	16	11
3	3	16	1	10	7	12	5	14
4	4	11	14	1	8	15	10	5
5	5	6	7	8	1	2	3	4
6	6	5	16	11	2	1	12	15
7	7	12	5	14	3	16	1	10
8	8	15	10	5	4	11	14	1

The group multiplication table can be displayed explicitly in terms of the group elements themselves.

mat = Map[Part[elm, #] &, tbl, {2}];								
TableForm[PauliForm@mat[[;; 8, ;; 8]],								
TableHeadings → PauliForm@{elm, elm}, TableAlignments → Right]								
Out[=]/TableForm=	I	X	Y	Z	-I	-X	-Y	-Z
I	I	X	Y	Z	-I	-X	-Y	-Z
X	X	I	i Z	-i Y	-X	-I	-i Z	i Y
Y	Y	-i Z	I	i X	-Y	i Z	-I	-i X
Z	Z	i Y	-i X	I	-Z	-i Y	i X	-I
-I	-I	-X	-Y	-Z	I	X	Y	Z
-X	-X	-I	-i Z	i Y	X	I	i Z	-i Y
-Y	-Y	i Z	-I	-i X	Y	-i Z	I	i X
-Z	-Z	-i Y	i X	-I	Z	i Y	-i X	I

Definition C.2 (generators) Let \mathcal{G} be a group. The elements G_1, G_2, \dots, G_k of \mathcal{G} are said to *generate* the group \mathcal{G} and denoted by

$$\mathcal{G} = \langle \{G_1, G_2, \dots, G_k\} \rangle \quad (\text{C.5})$$

if every element of \mathcal{G} can be expressed as a multiplication of them. The elements are called *generators* of \mathcal{G} .

The description of a group can be drastically simplified by the use of generators. For example, consider the Pauli group in (C.4). It contains 16 elements but is generated by three elements,

$$\mathcal{P}(1) = \langle \{X, Y, Z\} \rangle. \quad (\text{C.6})$$

It is more pronouncing when we consider the Pauli group $\mathcal{P}(2)$ on two qubits,

$$\mathcal{P}(2) = \{\pm I \otimes I, \pm i I \otimes I, \pm X \otimes I, \pm i X \otimes I, \pm X \otimes X, \pm i X \otimes X, \dots\}. \quad (\text{C.7})$$

There are 64 elements in $\mathcal{P}(2)$ whereas it is generated by just 6 elements

$$\mathcal{P}(2) = \langle \{X \otimes I, Y \otimes I, Z \otimes I, I \otimes X, I \otimes Y, I \otimes Z\} \rangle. \quad (\text{C.8})$$

For a given group \mathcal{G} , the set of generators is not unique. For example, the Pauli group on a single qubit can also be generated by iI , X , and Z ; $\mathcal{P}(1) = \langle \{iI, X, Z\} \rangle$.

Theorem C.3 Let \mathcal{G} be a finite group. There exists a set of $\log_2 |\mathcal{G}|$ elements that generates \mathcal{G} .

The description of a group can be drastically simplified by using generators.

```
In[5]:= gnr = GroupGenerators@grp;
PauliForm[gnr]
Out[5]= {X, Y, Z}
```

For another example, consider the Pauli group on two qubits.

```
In[6]:= elm = GroupElements@PauliGroup[S@{1, 2}];
PauliForm[elm]
Out[6]= {I⊗I, I⊗X, I⊗Y, I⊗Z, X⊗I, X⊗X, X⊗Y, X⊗Z, Y⊗I, Y⊗X, Y⊗Y, Y⊗Z,
Z⊗I, Z⊗X, Z⊗Y, Z⊗Z, -(I⊗I), -(I⊗X), -(I⊗Y), -(I⊗Z), -(X⊗I),
-(X⊗X), -(X⊗Z), -(Y⊗I), -(Y⊗X), -(Y⊗Y), -(Y⊗Z), -(Z⊗I),
-(Z⊗X), -(Z⊗Y), i I⊗I, i I⊗X, i I⊗Y, i I⊗Z, i X⊗I, i X⊗X,
i X⊗Y, i X⊗Z, i Y⊗I, i Y⊗X, i Y⊗Y, i Y⊗Z, i Z⊗I, i Z⊗X, i Z⊗Y, i Z⊗Z,
-i I⊗I, -i I⊗X, -i I⊗Y, -i I⊗Z, -i X⊗I, -i X⊗X, -i X⊗Y, -i X⊗Z,
-i Y⊗I, -i Y⊗X, -i Y⊗Y, -i Y⊗Z, -i Z⊗I, -i Z⊗X, -i Z⊗Y, -i Z⊗Z}
```

It has 64 elements. That is, the order of the Pauli group on two qubits is 64.

```
In[7]:= Length[elm]
Out[7]= 64
In[8]:= GroupOrder@PauliGroup[S@{1, 2}]
Out[8]= 64
```

The Pauli group on two qubits can be generated by just 6 elements.

```
In[9]:= gnr = GroupGenerators@PauliGroup[S@{1, 2}];
PauliForm[gnr]
Out[9]= {X⊗I, Y⊗I, Z⊗I, I⊗X, I⊗Y, I⊗Z}
```

Definition C.4 (Abelian group) If all the elements of a group commute, the group is said to be *Abelian*.

A special example of Abelian group is *cyclic groups*. A cyclic group \mathcal{G} consists of elements which can be obtained by raising one of them to successive powers, i.e.,

$$\mathcal{G} = \{G, G^2, G^3, \dots, G^n = E\}, \quad (\text{C.9})$$

where n is some integer. It is thus generated by a single element, $\mathcal{G} = \langle \{G\} \rangle$; Any element of a cyclic group generates the group. A common example of cyclic group is

$$\mathbb{Z}_n := \{0, 1, 2, \dots, n-1\} \quad (\text{C.10})$$

with addition modulo n as the group multiplication. In particular, \mathbb{Z}_2 appears very often in wide areas of science.

Theorem C.5 (*rearrangement theorem*) Let \mathcal{G} be a group. For any fixed element $G \in \mathcal{G}$, the set $GG := \{GG' \mid G' \in \mathcal{G}\}$ contains every element of \mathcal{G} *once and only once*. The same holds for the set $\mathcal{G}G := \{G'G \mid G' \in \mathcal{G}\}$.

Definition C.6 (*group homomorphism*) If $\phi : \mathcal{G} \rightarrow \mathcal{G}'$ is a mapping of a group \mathcal{G} onto another group \mathcal{G}' such that

$$\phi(G_1)\phi(G_2) = \phi(G_1G_2) \quad (\text{C.11})$$

for all $G_1, G_2 \in \mathcal{G}$, then ϕ is said to be a *homomorphic* mapping or a *homomorphism*. If ϕ is an one-to-one mapping, then it is called an *isomorphism*. When there exists an isomorphism from \mathcal{G} onto \mathcal{G}' , \mathcal{G} and \mathcal{G}' are said to be *isomorphic* to each other and denoted by $\mathcal{G} \simeq \mathcal{G}'$.

C.2 Classes

A *class* of a group \mathcal{G} is a subset of \mathcal{G} having a certain property. This particular property makes classes to play an important role in representation theory.

Definition C.7 (*conjugate elements and classes*) Let \mathcal{G} be a group. An element $G' \in \mathcal{G}$ is said to be *conjugate* to another element G if there exists an element $H \in \mathcal{G}$ such that

$$G' = HGH^{-1}. \quad (\text{C.12})$$

Obviously, if G' is conjugate to G , then G is also conjugate to G' . It is therefore permissible to talk of a set of *mutually* conjugate elements. A *class* of a group \mathcal{G} is a set of mutually conjugate elements of \mathcal{G} .

Each class is completely determined by any member G of it. That is, given G , we obtain the whole class by forming the products

$$\mathcal{G}GG^{-1} := \{HGH^{-1} \mid H \in \mathcal{G}\}. \quad (\text{C.13})$$

Theorem C.8 Classes have the following properties:

- (a) Every element of a group \mathcal{G} is a member of some class of \mathcal{G} .
- (b) No element of \mathcal{G} can be a member of two different classes of \mathcal{G} .
- (c) The identity E of \mathcal{G} always forms a class on its own.

Conjugation relation between group elements as defined in (C.12) has physical implications when applied to quantum mechanics. For example, if \mathcal{G} is a group consisting of rotations, no class of \mathcal{G} contains both proper and improper rotations. Moreover, in each class of proper rotations all the rotations are through the same angle. Similarly, in each class of improper rotations the proper parts are all through the same angle.

C.3 Invariant Subgroups

The notion of invariant subgroups is usually introduced together with the notion of *cosets* to construct *factor groups*. It is also closely related to the concept of *classes* as will be seen shortly. Therefore, invariant subgroups also play an important role in representation theory.

Definition C.9 (*subgroup*) Let \mathcal{G} be a group and $\mathcal{H} \subset \mathcal{G}$. If \mathcal{H} itself is a group (with the multiplication among its elements given by that of \mathcal{G}), then \mathcal{H} is called a subgroup of the group \mathcal{G} .

Definition C.10 (*invariant subgroups*) A subgroup \mathcal{H} of a group \mathcal{G} is said to be *invariant* if

$$GHG^{-1} \in \mathcal{H} \quad (\text{C.14})$$

for every $H \in \mathcal{H}$ and $G \in \mathcal{G}$. In many cases the property in Eq. (C.14) is written in a more compact form as

$$\mathcal{G}\mathcal{H}\mathcal{G}^{-1} = \mathcal{H} \quad (\text{C.15})$$

with the abbreviation $\mathcal{G}\mathcal{H}\mathcal{G}^{-1} := \{GHG^{-1} \mid H \in \mathcal{H}, G \in \mathcal{G}\}$.

Invariant subgroups are sometimes called *normal subgroups* or *normal divisors* because the defining property in Eq. (C.14) is of the same form as Eq. (C.12). The following theorem ensures the connection between the *classes* and the invariant subgroups.

Theorem C.11 A subgroup \mathcal{H} of a group \mathcal{G} is an invariant subgroup if and only if \mathcal{H} consists entirely of *complete* classes of \mathcal{G} .

A special kind of invariant subgroup is the *centre* of the group. This notion is closely related to the notion of *factor groups*.

Definition C.12 (*centre of a group*) The *centre* \mathcal{Z} of a group \mathcal{G} is defined to be the subgroup consisting of *all* elements of \mathcal{G} that commute with every element of \mathcal{G} .

\mathcal{Z} is an Abelian invariant subgroup of \mathcal{G} . Moreover, any subgroup of \mathcal{Z} is an Abelian invariant subgroup of \mathcal{G} and called a *central invariant subgroup* of \mathcal{G} .

C.4 Cosets and Factor Groups

A subgroup of a group can be used to decompose the group (as a set) into disjoint subsets called *cosets*. We have already seen in Section C.2 that conjugacy classes is another way to do it. However, unlike conjugacy classes, all cosets of a group is of equal size.

Definition C.13 (coset) Let \mathcal{H} be a subgroup (not necessarily an invariant subgroup) of a group \mathcal{G} . Then for any fixed $G \in \mathcal{G}$ (which may or may not be member of \mathcal{H}) the set $\mathcal{H}G := \{HG \mid H \in \mathcal{H}\}$ is called the *right coset* of \mathcal{H} with respect to G . Similarly, the set $G\mathcal{H} := \{GH \mid H \in \mathcal{H}\}$ is called the *left coset* of \mathcal{H} with respect to G .

The properties of cosets are summarized in the following two theorems:

Theorem C.14 Let \mathcal{H} be a subgroup of a group \mathcal{G} .

- (a) If $G \in \mathcal{H}$, then $\mathcal{H}G = \mathcal{H}$.
- (b) If $G' \in \mathcal{H}G$, then $\mathcal{H}G' = \mathcal{H}G$.
- (c) If $G \notin \mathcal{H}$, then $\mathcal{H}G$ is not a subgroup of \mathcal{G} .
- (d) Every element of \mathcal{G} is a member of some right coset.
- (e) Two right cosets of \mathcal{H} are either identical or have no elements in common.
- (f) Any two elements HG and $H'G$ of $\mathcal{H}G$ are different provided that $H \neq H'$.
In particular, if \mathcal{H} is a finite subgroup of order $|\mathcal{H}|$, then $\mathcal{H}G$ contains $|\mathcal{H}|$ different elements.
- (g) If \mathcal{G} is a finite group of order $|\mathcal{G}|$ and \mathcal{H} has order $|\mathcal{H}|$, then the number of distinct right cosets is $|\mathcal{G}|/|\mathcal{H}|$.

Above the statements are for the right cosets, but every statement applies equally to left cosets.

Theorem C.15 (cosets and invariant subgroups) The right and left cosets of a subgroup \mathcal{H} of a group \mathcal{G} are *identical* (i.e., $\mathcal{H}G = G\mathcal{H}$ for all $G \in \mathcal{G}$) if and only if \mathcal{H} is an *invariant* subgroup.

The property (b) of Theorem C.14 is particularly important. It shows that the same coset is formed starting from *any* member of the coset. All members of a coset therefore appear on an equal footing, so that *any* member of the coset can be taken as the *coset representative* that labels the coset and from which the coset can be constructed. Accordingly, it may be useful to ignore the internal structure of each right coset of a subgroup \mathcal{H} , and consider each coset as an *element* of the *set of distinct right cosets*. Indeed, such a set of cosets forms a group with the suitable definition of multiplication as stated in the following theorem:

Theorem C.16 (factor group \mathcal{G}/\mathcal{H}) The set of right cosets of an *invariant* subgroup \mathcal{H} of a group \mathcal{G} forms a group with the group multiplication defined by

$$G_1\mathcal{H} \cdot G_2\mathcal{H} = (G_1G_2)\mathcal{H} \quad (\text{C.16})$$

for any $G_1, G_2 \in \mathcal{G}$. This group is called a *factor (or quotient) group* and is denoted by \mathcal{G}/\mathcal{H} . Further, if \mathcal{G} is *finite*, then^{C.1}

$$|\mathcal{G}/\mathcal{H}| = |\mathcal{G}|/|\mathcal{H}|. \quad (\text{C.17})$$

The key idea behind the notion of factor group is that two elements G, G' of \mathcal{G} are regarded equivalent as long as there exists $H \in \mathcal{H}$ such that $G = G'H$ (or $H' \in \mathcal{H}$ such that $G = H'G'$).

Note that unless the subgroup \mathcal{H} is an invariant subgroup, it is not guaranteed that Eq. (C.16) provides a meaningful and consistent definition for the group multiplication. One has first to show that Eq. (C.16) provides a meaningful definition in that if alternative coset representatives are chosen for the cosets on the left-hand side of the equation, the coset on the right-hand side remains unchanged. Suppose that G'_1 and G'_2 are alternative coset representatives for $\mathcal{H}G_1$ and $\mathcal{H}G_2$, respectively, so that $G'_1 \in \mathcal{H}G_1$ and $G'_2 \in \mathcal{H}G_2$. It has to be proved that $\mathcal{H}(G'_1G'_2) = \mathcal{H}(G_1G_2)$. As $G'_1 \in \mathcal{H}G_1$ and $G'_2 \in \mathcal{H}G_2$, there exist $H_1, H_2 \in \mathcal{H}$ such that $G'_1 = H_1G_1$ and $G'_2 = H_2G_2$. Further, as \mathcal{H} is invariant, $G_1\mathcal{H} = \mathcal{H}G_1$ and hence there exists $H'_2 \in \mathcal{H}$ such that $G_1H_2 = H'_2G_1$. Consequently, one has

$$\mathcal{H}G'_1 \cdot \mathcal{H}G'_2 = \mathcal{H}(H_1G_1H_2G_2) = \mathcal{H}(H_1H'_2G_1G_2) = \mathcal{H}(G_1G_2). \quad (\text{C.18})$$

As an example, consider again the Pauli group on a single qubit

$$\mathcal{G} = \{\pm I, \pm iI, \pm X, \pm iX, \pm Y, \pm iY, \pm Z, \pm iZ\}, \quad (\text{C.19})$$

which we have already discussed in Eq. (C.4). $\mathcal{Z} = \{I, -I, iI, -iI\}$ is an invariant subgroup of \mathcal{G} . In fact, \mathcal{Z} is the center of \mathcal{G} —see Definition C.12. The factor group \mathcal{G}/\mathcal{Z} is given by

$$\mathcal{G}/\mathcal{Z} = \{\mathcal{Z}, X\mathcal{Z}, Y\mathcal{Z}, Z\mathcal{Z}\}. \quad (\text{C.20})$$

In the factor group \mathcal{G}/\mathcal{Z} , the elements $X, -X, iX$, and $-iX$ are not distinguished and regarded as equivalent. Likewise, the elements $Y, -Y, iY$, and $-iY$ are regarded equivalent. This is convenient, for example, when one only interested in the commutation relation but not the explicit multiplications between elements.

^{C.1}It follows from Theorem C.14 (g).

C.5 Product Groups

Given two or more sets, it is common to construct a new set consisting of tuples of the elements from the given sets. Given two or more groups, one can construct a new group in an analogous manner with the group multiplication inherited from the given groups.

Definition C.17 Let \mathcal{G} and \mathcal{G}' be groups. The *direct product group* or simply *product group*, $\mathcal{G} \times \mathcal{G}'$, is defined as follows:

- (a) The underlying set of $\mathcal{G} \otimes \mathcal{G}'$ consists of the ordered pair of elements from \mathcal{G} and \mathcal{G}' , that is,

$$\mathcal{G} \otimes \mathcal{G}' := \{(G, G') \mid G \in \mathcal{G}, G' \in \mathcal{G}'\} \quad (\text{C.21})$$

- (b) For any $G_1, G_2 \in \mathcal{G}$ and $G'_1, G'_2 \in \mathcal{G}'$, the group multiplication is defined by

$$(G_1, G'_1)(G_2, G'_2) = (G_1 G_2, G'_1 G'_2). \quad (\text{C.22})$$

It is straightforward to extend the definition to direct products of more than two groups by repeating the above operation.

The simplest example is the direct product of cyclic groups [see Eq. (C.10)], $\mathbb{Z}_{n_1} \times \mathbb{Z}_{n_2} \times \cdots \times \mathbb{Z}_{n_k}$. The resulting group is Abelian—see Definition C.4. In fact, one can show that any Abelian group is isomorphic to $\mathbb{Z}_{n_1} \times \mathbb{Z}_{n_2} \times \cdots \times \mathbb{Z}_{n_k}$ for some n_1, n_2, \dots, n_k . Interestingly, $\mathbb{Z}_{n_1} \times \mathbb{Z}_{n_2} \times \cdots \times \mathbb{Z}_{n_k}$ can also be regarded as a vector space over the field \mathbb{Z}_2 . This fact provides convenient tools in studying the structure of Pauli group—see Section 6.3.1.

Appendix D

Mathematica Application Q3

• August 5, 2021 (v1.7)

Q3 is a Mathematica application to help study quantum information processing, quantum many-body systems, and quantum spin systems. It provides various tools and utilities for symbolic calculations and numerical simulations in these areas of quantum physics.

Q3 consists of several packages at different levels. Quisso, Fock, and Wigner are the three main packages, and they are devoted to the simulation of quantum information processing, quantum many-body systems, and quantum spin systems, respectively. They are based on another lower-level package, Pauli. Pauli itself provides useful tools to handle the Pauli matrices and operators for unlabelled qubits directly. But it also lays out the programming structures and defines objects for the aforementioned three and other higher-level packages.

Q3 is distributed through the GitHub repository:

<https://github.com/quantum-mob/Q3App>

D.1 Installation

Q3 provides two installation methods: The first is based on the paclet system that has recently been introduced by Wolfram Research. It is not only fully automatic but also convenient to get updates later on. But it is supported only for Mathematica 12.1 or later. If your copy of Mathematica is compatible, then just copy the following code and run it on your Mathematica Notebook:

```
Module[
  {ps},
  ps = PacletSiteRegister[
    "https://github.com/quantum-mob/PacletServer/raw/main",
    "Quantum Mob Paclet Server"]
```

```
];
PacletSiteUpdate[ps];
PacletInstall["Q3"]
]
```

The other method is to download and copy the files to a proper folder—just the traditional method. For details, take a look at the *installation guide* at:

<https://github.com/quantum-mob/Q3App/blob/main/INSTALL.md>

D.2 Quick Start

Once the application is installed, put

```
"Q3" or "Q3/guide/Q3"
```

in the search field of the Wolfram Language Documentation Center (the help window of Mathematica) to get detailed technical information about the application.

Note that after installing the application, the first time you search a keyword in Wolfram Language Documentation Center, Mathematica builds the search index of the new documentation files. It can take a few seconds to minutes depending on your computer. It happens only once (everytime you update the application).

Appendix E

Integrated Compilation of Demonstrations

`QuantumWorkbook` is a compilation of Mathematica Notebook files containing the Wolfram Language codes that have been used to generate the demonstrations in the book. Readers can try and modify the codes themselves to build their own examples on the demonstrations and to experiment fresh ideas.

`QuantumWorkbook` is distributed through the GitHub repository:

<https://github.com/quantum-mob/QuantumWorkbook>

E.1 Installation

Copy the following code, and just evaluate it in your Mathematica(R) Notebook:

```
Module[
  { ps },
  ps = PacletSiteRegister[
    "https://github.com/quantum-mob/PacletServer/raw/main",
    "Quantum Mob Paclet Server"
  ];
  PacletSiteUpdate[ps];
  PacletInstall["Q3"];
  PacletInstall["QuantumWorkbook"]
]
```

Note that along with `QuantumWorkbook`, it also installs the main application Q3 for your convenience.

This package may be modified for bug fixes and improvements. You may want to check for updates from time to time:

```
QuantumWorkbookCheckUpdate[]
```

In case there is an update, you can install it by using the following function:

```
QuantumWorkbookUpdate[]
```

E.2 Quick Start

Once `QuantumWorkbook` is installed, put

```
"QuantumWorkbook"
```

in the search field of the Wolfram Documentation Center (the Help window of Mathematica). You will see the table of contents of the workbook.

Appendix F

Solutions to Select Problems

- August 3, 2021 (v1.4)

F.1 The Postulates of Quantum Mechanics

F.2 Quantum Computation: Overview

Problem 2.4 First, note that

$$\begin{array}{c} \text{---} \\ | \quad \bullet \quad | \\ \boxed{H} \quad \bigoplus \quad \boxed{H} \end{array} = \begin{array}{c} \text{---} \\ | \quad \bullet \quad | \\ \boxed{Z} \end{array} = \begin{array}{c} \text{---} \\ | \quad | \quad | \\ & \bullet \quad | \end{array},$$

where note that the controlled-Z operation is “symmetric” in the roles of control and target; hence the circuit representation by two dots. Then it follows that

$$\begin{array}{c} \text{---} \\ | \quad \bullet \quad | \\ \boxed{H} \quad \bigoplus \quad \boxed{H} \\ | \quad \bullet \quad | \\ \boxed{H} \quad \bigoplus \quad \boxed{H} \end{array} = \begin{array}{c} \text{---} \\ | \quad \bullet \quad | \\ \boxed{H} \quad \bigoplus \quad \boxed{H} \\ | \quad \bullet \quad | \\ \boxed{Z} \end{array} = \begin{array}{c} \text{---} \\ | \quad \bullet \quad | \\ \boxed{H} \quad \bigoplus \quad \boxed{H} \\ | \quad | \quad | \\ & \bullet \quad | \end{array} \\ = \begin{array}{c} \text{---} \\ | \quad \bullet \quad | \\ \boxed{H} \quad \bigoplus \quad \boxed{H} \\ | \quad \bullet \quad | \\ \boxed{H} \quad \bigoplus \quad \boxed{H} \end{array} = \begin{array}{c} \text{---} \\ | \quad \bullet \quad | \\ \bigoplus \end{array}. \quad (\text{F.1})$$

Problem 2.7

The eigenstates of U are the same as those of the Pauli X operator.

```
In[7]:= U = Rotation[\phi, S[1, 1]] // Elaborate
```

```
Out[7]= Cos[\frac{\phi}{2}] - I Sin[\frac{\phi}{2}]
```

To see it, take an eigenstate of the Pauli X operator on qubit `s[1, None]` belonging to the eigenvalue 1.

```
In[7]:= vec = S[1, 6] ** Ket[];
vec // LogicalForm
```

$$\frac{|\Theta_{S_1}\rangle}{\sqrt{2}} + \frac{|1_{S_1}\rangle}{\sqrt{2}}$$

This confirms that the vector is indeed the intended eigenstate.

```
In[8]:= U ** vec // LogicalForm // TrigToExp // Simplify
```

$$\frac{e^{-\frac{i\phi}{2}} (|\Theta_{S_1}\rangle + |1_{S_1}\rangle)}{\sqrt{2}}$$

This checks for the other eigenstate belonging to the eigenvalue -1.

```
In[9]:= vec = S[1, 6] ** Ket[S[1] \rightarrow 1];
vec // LogicalForm
```

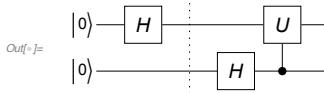
$$\frac{|\Theta_{S_1}\rangle}{\sqrt{2}} - \frac{|1_{S_1}\rangle}{\sqrt{2}}$$

```
In[10]:= U ** vec // LogicalForm // TrigToExp // Simplify
```

$$\frac{e^{\frac{i\phi}{2}} (|\Theta_{S_1}\rangle - |1_{S_1}\rangle)}{\sqrt{2}}$$

The ancillary qubit takes a relative phase shift depending on in which eigenstate the native qubit is.

```
In[11]:= Let[Real, \phi];
qc = QuantumCircuit[LogicalForm[Ket[], S@{1, 2}], S[1, 6], "Separator",
S[2, 6], ControlledU[S[2], Rotation[\phi, S[1, 1]]], "Label" \rightarrow "U"]
```

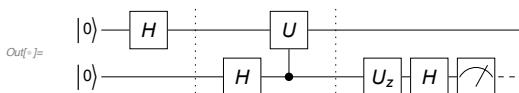


```
In[12]:= out = ExpressionFor[qc] // TrigToExp;
LogicalForm[QuissoFactor@out, S@{1, 2}]
```

$$\frac{1}{2} e^{-\frac{i\phi}{2}} (|\Theta_{S_1}\rangle + |1_{S_1}\rangle) \otimes \left(e^{\frac{i\phi}{2}} |\Theta_{S_2}\rangle + |1_{S_2}\rangle \right)$$

Make a basis change to detect it.

```
In[13]:= qc = QuantumCircuit[LogicalForm[Ket[], S@{1, 2}], S[1, 6], "Separator",
S[2, 6], ControlledU[S[2], Rotation[\phi, S[1, 1]]], "Label" \rightarrow "U"],
"Separator", Rotation[\phi/2, S[2, 3]], S[2, 6], Measurement[S[2]]]
```



Finally, check the result.

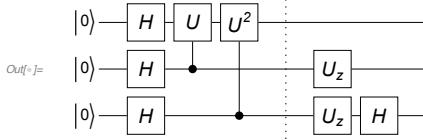
```
In[14]:= out = ExpressionFor[qc] // TrigToExp;
LogicalForm[QuissoFactor@out, S@{1, 2}]
```

$$\frac{e^{-\frac{i\phi}{4}} (|\Theta_{S_1}\Theta_{S_2}\rangle + |1_{S_1}1_{S_2}\rangle)}{\sqrt{2}}$$

Problem 2.8

This constructs the desired quantum circuit model.

```
In[1]:= qc = QuantumCircuit[LogicalForm[Ket[], S@{1, 2, 3}],  
S[{1, 2, 3}, 6], ControlledU[S[2], Rotation[\phi, S[1, 1]], "Label" \rightarrow "U"],  
ControlledU[S[3], Rotation[2\phi, S[1, 1]], "Label" \rightarrow "U^2"],  
"Separator", {Rotation[0, S[2, 3]], Rotation[\Pi/2, S[3, 3]]}, S[3, 6]]
```



```
In[2]:= out = ExpressionFor[qc] // TrigToExp // Simplify;  
LogicalForm[QuissoFactor@out /. \phi \rightarrow \Pi/2, S@{1, 2}]  
Out[2]= 
$$\frac{\left(\frac{1}{4} + \frac{i}{4}\right) e^{-\frac{3i\pi}{4}} (\left|0_{S_1}\right\rangle + \left|1_{S_1}\right\rangle) \otimes \left(e^{\frac{i\pi}{4}} \left|0_{S_2}\right\rangle + \left|1_{S_2}\right\rangle\right) \otimes (2 \left|0_{S_3}\right\rangle)}{\sqrt{2}}$$

```

Problem 2.10 Suppose that the statement (2.10a) holds. Then

$$\hat{A}\hat{B} = \hat{I}, \quad \hat{A}\hat{X}\hat{B} = \hat{U}. \quad (\text{F.2})$$

The first condition in the above implies that $\hat{B} = \hat{A}^\dagger$. Then put $\hat{W} = \hat{A}\hat{H}$, where \hat{H} is the Hadamard gate, so that

$$\hat{W}\hat{Z}\hat{W}^\dagger = \hat{A}\hat{X}\hat{A}^\dagger = \hat{U}. \quad (\text{F.3})$$

That is, the statement (2.10b) holds. Further, observe that

$$\text{Tr } \hat{U} = \text{Tr } \hat{W}\hat{Z}\hat{W}^\dagger = \text{Tr } \hat{Z} = 0, \quad (\text{F.4})$$

$$\det \hat{U} = \det \hat{W}\hat{Z}\hat{W}^\dagger = \det \hat{Z} = -1, \quad (\text{F.5})$$

and hence the statement (2.10c) holds as well.

Next, suppose that $\hat{U} = \hat{W}\hat{Z}\hat{W}^\dagger$ for some unitary operator \hat{W} . Put $\hat{A} = \hat{W}\hat{H}$ and $\hat{B} = \hat{A}^\dagger$, which satisfy (F.2); that is, the statement (2.10a) holds. Further, statement (2.10c) also holds as already shown above.

Finally, suppose that the statement (2.10c) holds: The condition $\text{Tr } \hat{U} = 0$ implies that

$$\hat{U} = u_x\hat{X} + u_y\hat{Y} + u_z\hat{Z} \quad (u_x, u_y, u_z \in \mathbb{C}). \quad (\text{F.6})$$

The unitarity condition of \hat{U} implies that all the phases of u_μ must be the same and that $\sum_\mu |u_\mu|^2 = 1$. This implies in turn that

$$\hat{U} = e^{i\phi}\hat{W}\hat{Z}\hat{W}^\dagger \quad (\text{F.7})$$

for some $\phi \in \mathbb{R}$ and some unitary operator \hat{W} . The condition $\det \hat{U} = -1$ leads to $\phi = n\pi$ with $n \in \mathbb{Z}$. For $e^{i\phi} = -1$ (n odd), simply replace \hat{W} with $\hat{W}' = \hat{W}\hat{X}$. Therefore, regardless of $e^{i\phi} = \pm 1$, the statement (2.10b) holds. As we have already shown above that the statement (2.10b) implies (2.10a), this completes the proof.

F.3 Quantum Computers

Problem 3.3 We define

$$|D\rangle := \frac{|1\rangle\Omega_1 + |2\rangle\Omega_2}{\Omega}. \quad (\text{F.8})$$

where $\Omega := \sqrt{\Omega_1^2 + \Omega_2^2}$. Then the Hamiltonian reads as

$$\hat{H} = \epsilon |\epsilon\rangle\langle\epsilon| - \frac{1}{2}\Omega (|\Omega\rangle\langle\epsilon| + |\epsilon\rangle\langle\Omega|). \quad (\text{F.9})$$

- (a) The expressions for the two eigenstates of \hat{H} in (F.9) are formally the same as those in the main text (Section 3.3.1). The Hamiltonian involves only $|\epsilon\rangle$ and $|\Omega\rangle$. There must be one more state. It is the dark state of the system. We denote it by $|D\rangle$. The dark state must be orthogonal to the bright state $|\Omega\rangle$ as well as $|\epsilon\rangle$. Therefore, we can construct it as follows

$$|D\rangle = \frac{|1\rangle\Omega_2^* - |2\rangle\Omega_1^*}{\Omega}. \quad (\text{F.10})$$

By inspection, one can confirm that it is orthogonal both to $|\Omega\rangle$ and trivially to $|\epsilon\rangle$. Using the parameterization suggested in the statement of the problem, we can rewrite $|D\rangle$ into the form

$$|D\rangle = \frac{|1\rangle \sin(\theta/2)e^{-\phi/2} - |2\rangle \cos(\theta/2)e^{i\phi/2}}{\Omega}. \quad (\text{F.11})$$

- (b) The Abelian gauge potential is given by

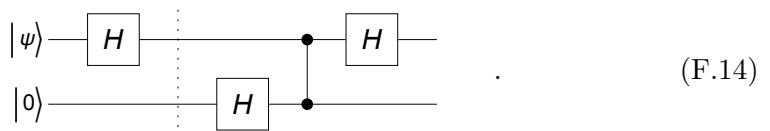
$$A^\phi := \langle D | \frac{\partial}{\partial \phi} | D \rangle = \frac{i \cos \theta}{2}. \quad (\text{F.12})$$

As it is Abelian, it is customary to define the Berry phase as $\gamma = -iA^\phi = \frac{1}{2}\cos\theta$.

- (c) For the path where ϕ varies from 0 to 2π with θ fixed, the Abelian geometric phase is given by

$$U(\mathcal{C}) = e^{-i\gamma} = e^{-i(\cos\theta)/2}. \quad (\text{F.13})$$

Problem 3.4 A direct inspection would be sufficient. Here we slightly modify the quantum circuit model into the form



We then use the already known result in (3.70) for the quantum circuit model in Eq. (3.69). In this case, \hat{U}_z is trivial, $\hat{U}_z = \hat{I}$, and the input state is $\hat{H}|\psi\rangle$. It leads to the output state on the second qubit

$$\frac{|0\rangle \otimes (\hat{H}\hat{H}|\psi\rangle) + |1\rangle \otimes (\hat{H}\hat{Z}\hat{H}|\psi\rangle)}{\sqrt{2}}. \quad (\text{F.15})$$

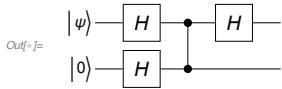
The identities, $\hat{H}^2 = \hat{I}$ and $\hat{H}\hat{Z}\hat{H} = \hat{X}$, lead to

$$\frac{|0\rangle \otimes |\psi\rangle + |1\rangle \otimes (\hat{X}|\psi\rangle)}{\sqrt{2}}. \quad (\text{F.16})$$

as expected.

Here is a quantum circuit model to implement the Pauli X gate based on measurement.

```
In[7]:= qc = QuantumCircuit[ProductState[S[1] → {c[0], c[1]}, "Label" → Ket["ψ"]], LogicalForm[Ket[], S@{2}], S[{1, 2}, 6], CZ[S[1], S[2]], S[1, 6]]
```

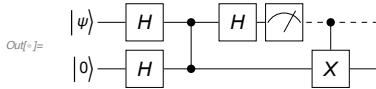


```
In[8]:= out = Elaborate[qc];
QuissoFactor[out, S[1]]
```

$$\text{Out[8]}= \left|0_{S_1}\right\rangle \otimes \left(\frac{c_0 \left|-\right\rangle}{\sqrt{2}} + \frac{c_1 \left|1_{S_2}\right\rangle}{\sqrt{2}}\right) + \left|1_{S_1}\right\rangle \otimes \left(\frac{c_1 \left|-\right\rangle}{\sqrt{2}} + \frac{c_0 \left|1_{S_2}\right\rangle}{\sqrt{2}}\right)$$

We go further to post-process the output state on the second qubit.

```
In[9]:= qc1 = QuantumCircuit[qc, Measurement[S[1]], ControlledU[S[1], S[2, 1]]]
```



```
In[10]:= new = Elaborate[qc1] /. {Conjugate[c[0]] × c[0] + Conjugate[c[1]] × c[1] → 1};
LogicalForm[QuissoFactor[new, S[1]], S@{1, 2}]
```

$$\text{Out[10]}= \left|0_{S_1}\right\rangle \otimes (c_0 \left|0_{S_2}\right\rangle + c_1 \left|1_{S_2}\right\rangle)$$

Problem 3.6 Let \hat{U} be a unitary operator on a finite-finite dimensional vector space \mathcal{V} . Then we have

$$\hat{U} \exp(\hat{A}) \hat{U}^\dagger = \exp(\hat{U} \hat{A} \hat{U}^\dagger) \quad (\text{F.17})$$

for any linear operator \hat{A} on \mathcal{V} .

- (a) We note that \hat{X} is both unitary and Hermitian. Further, $\hat{X}\hat{Z}\hat{X} = -\hat{Z}$. These observations lead to

$$\hat{X} \exp(-i\hat{Z}\phi/2) \hat{X} = \exp(-i\hat{X}\hat{Z}\hat{X}\phi/2) = \exp(+i\hat{Z}\phi/2) = \hat{U}_z(-\phi). \quad (\text{F.18})$$

- (b) Similarly, \hat{H} is both unitary and Hermitian. It also satisfies $\hat{H}\hat{Z}\hat{H} = \hat{X}$. It follows that

$$\hat{H} \exp(-i\hat{Z}\phi/2)\hat{H} = \exp(-i\hat{H}\hat{Z}\hat{H}\phi/2) = \exp(-i\hat{X}\phi/2) = \hat{U}_x(\phi). \quad (\text{F.19})$$

F.4 Quantum Algorithms

Problem 4.1

Here is a particular classical oracle.

```
In[1]:= f[0, 0] = 0
f[0, 1] = 0
f[1, 0] = 1
f[1, 1] = 0

Out[1]= 0

In[2]:= f[0]
Out[2]= 0

In[3]:= f[1]
Out[3]= 1

In[4]:= f[0]
Out[4]= 0

In[5]:= cc = {1, 2};
tt = {3};
ct = Join[cc, tt];
qc = QuantumCircuit[
  LogicalForm[Ket[S[tt] → 1], S[ct]],
  S[ct, 6],
  Oracle[f, S@cc, S@tt],
  S[tt, 6]
]
out = ExpressionFor[qc];
QuissoFactor[out, S[tt]] // LogicalForm

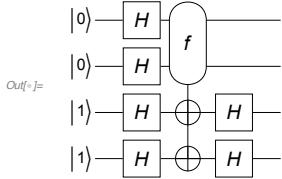
Out[5]= |1S3⟩ ⊗ (1/2 |0S1 0S2⟩ + 1/2 |0S1 1S2⟩ - 1/2 |1S1 0S2⟩ + 1/2 |1S1 1S2⟩)
```

Here is a particular classical oracle.

```
In[n]:= f[0, 0] = {0, 1}
          f[0, 1] = {1, 0}
          f[1, 0] = {0, 1}
          f[1, 1] = {1, 1}

Out[n] = {0, 1}
Out[n] = {1, 0}
Out[n] = {0, 1}
Out[n] = {1, 1}

In[n]:= cc = {1, 2};
          tt = {3, 4};
          ct = Join[cc, tt];
          qc = QuantumCircuit[
          LogicalForm[Ket[S[tt] → 1], S[ct]],
          S[ct, 6],
          Oracle[f, S@cc, S@tt],
          S[tt, 6]
        ]
          out = ExpressionFor[qc];
          QuissoFactor[out, S[tt]] // LogicalForm
```



$$\text{Out}[n] = \left| 1_{S_3} 1_{S_4} \right\rangle \otimes \left(-\frac{1}{2} \left| 0_{S_1} 0_{S_2} \right\rangle - \frac{1}{2} \left| 0_{S_1} 1_{S_2} \right\rangle - \frac{1}{2} \left| 1_{S_1} 0_{S_2} \right\rangle + \frac{1}{2} \left| 1_{S_1} 1_{S_2} \right\rangle \right)$$

```
In[n]:= bb = f @@ IntegerDigits[Range[0, 2^2], 2, 2]
```

```
Out[n] = {{0, 1}, {1, 0}, {0, 1}, {1, 1}, {0, 1}}
```

F.5 Decoherence

Problem 5.1

F.6 Quantum Error-Correction Codes

Problem 6.3 Let $|\hat{G}_1\rangle, |\hat{G}_2\rangle, \dots, |\hat{G}_k\rangle$ be the Gottesman vectors of the generators $\hat{G}_1, \hat{G}_2, \dots, \hat{G}_k$. As the generators are independent, the Gottesman vectors are linearly independent of each other. Construct a matrix M from the rows of $\langle \hat{G}_1|, \langle \hat{G}_2|, \dots, \langle \hat{G}_k|$,

$$M = \begin{bmatrix} \langle \hat{G}_1| \\ \langle \hat{G}_2| \\ \vdots \\ \langle \hat{G}_k| \end{bmatrix} = \begin{bmatrix} M_{11} & M_{12} & \cdots & M_{1n} \\ & \vdots & & \\ M_{k1} & M_{k2} & \cdots & M_{kn} \end{bmatrix}. \quad (\text{F.20})$$

Suppose that we want to find an operator \hat{G} that anti-commutes with, say, \hat{G}_1 but commute with all others. Because the rows of M are linearly independent by construction, the equation

$$\begin{bmatrix} M_{11} & M_{12} & \cdots & M_{1n} \\ \vdots & & & \\ M_{k1} & M_{k2} & \cdots & M_{kn} \end{bmatrix} \begin{bmatrix} y_1 \\ y_2 \\ y_3 \\ \vdots \\ y_n \end{bmatrix} = \begin{bmatrix} 1 \\ 0 \\ 0 \\ \vdots \\ 0 \end{bmatrix}. \quad (\text{F.21})$$

has at least one solution. Define a Gottesman vector $|\hat{G}\rangle := (y_1, y_2, y_3, \dots, y_n)\hat{J}$ where the operator \hat{J} on the Gottesman vector space is defined in Eq. (6.43). Then we have

$$\begin{bmatrix} M_{11} & M_{12} & \cdots & M_{1n} \\ \vdots & & & \\ M_{k1} & M_{k2} & \cdots & M_{kn} \end{bmatrix} \hat{J} |\hat{G}\rangle = \begin{bmatrix} 1 \\ 0 \\ 0 \\ \vdots \\ 0 \end{bmatrix}. \quad (\text{F.22})$$

That is,

$$\langle \hat{G}_1 | \hat{G} \rangle = 1, \quad \langle \hat{G}_2 | \hat{G} \rangle = 0, \quad \dots, \quad \langle \hat{G}_k | \hat{G} \rangle = 0. \quad (\text{F.23})$$

Therefore, the operator \hat{G} corresponding to the Gottesman vector $|\hat{G}\rangle$ must anti-commute with \hat{G}_1 but commute with $\hat{G}_2, \dots, \hat{G}_k$.

Problem 6.5 See Problem 2.4.

Problem 6.6 Let

$$\hat{Z}' := \hat{U}(\hat{Z} \otimes \hat{J})\hat{U}^\dagger, \quad \hat{X}' := \hat{U}(\hat{X} \otimes \hat{J})\hat{U}^\dagger. \quad (\text{F.24})$$

Note that they anti-commute. Therefore, there exists at least one qubit on which \hat{Z}' and \hat{X}' anti-commute. We rearrange the qubits so that the particular qubit comes at the first place. Now \hat{Z}' and \hat{X}' must be of the form

$$\hat{Z}' = \hat{P} \otimes \hat{A}, \quad \hat{X}' = \hat{Q} \otimes \hat{B} \quad (\text{F.25})$$

with $\{\hat{P}, \hat{Q}\} = 0$. Then, one can apply the conjugation by a single-qubit operation in $\mathcal{C}(1)$ on the first qubit so that

$$\hat{P} \rightarrow \hat{X}, \quad \hat{Q} \rightarrow \hat{Z}. \quad (\text{F.26})$$

Problem 6.7 Let \hat{P} be an element of the n -qubit Pauli group $\mathcal{P}(n)$. Consider

$$\begin{aligned}\hat{P}\hat{V}|y\rangle &= \sum_x \hat{P}|x\rangle (\langle 0| \otimes \langle x|)\hat{U}(|0\rangle \otimes |y\rangle) \\ &= \sum_x \hat{P}|x\rangle (\langle 0| \otimes \langle x|)\hat{P}^\dagger(\hat{I} \otimes \hat{P})\hat{U}(|0\rangle \otimes |y\rangle) \\ &= \sum_x |x\rangle (\langle 0| \otimes \langle x|)(\hat{I} \otimes \hat{P})\hat{U}(|0\rangle \otimes |y\rangle),\end{aligned}$$

where we have used the fact that \hat{P} is invertible, and accordingly changed the dummy variable $\hat{P}|x\rangle \rightarrow |x\rangle$. Now as \hat{U} is an element of the Clifford group and $\hat{I} \otimes \hat{P} \in \mathcal{P}(n+1)$, there must exist $\hat{P}' \in \mathcal{P}(n+1)$ such that $(\hat{I} \otimes \hat{P})\hat{U} = \hat{U}\hat{P}'$.

$$\hat{P}\hat{V}|y\rangle = \sum_x |x\rangle (\langle 0| \otimes \langle x|)\hat{U}\hat{P}'(|0\rangle \otimes |y\rangle).$$

(i) Suppose that $\hat{P}' = \hat{Z} \otimes \hat{P}''$ or $\hat{P}' = \hat{I} \otimes \hat{P}''$ for some $\hat{P}'' \in \mathcal{P}(n)$. Then,

$$\begin{aligned}\hat{P}\hat{V}|y\rangle &= \sum_x |x\rangle (\langle 0| \otimes \langle x|)\hat{U}(|0\rangle \otimes \hat{P}''|y\rangle) \\ &= \hat{V}\hat{P}''|y\rangle\end{aligned}$$

for all $|y\rangle$. Therefore, $\hat{V} \in \mathcal{C}(n)$. (ii) Next, suppose that $\hat{P}' = \hat{X} \otimes \hat{P}''$; the case $\hat{P}' = \hat{Y} \otimes \hat{P}''$ can be treated by combining with the above argument. Then, using the property

$$\hat{U}(\hat{X} \otimes \hat{J})\hat{U}^\dagger = \hat{Z} \otimes \hat{P}''', \quad \hat{J} := \hat{I}^{\otimes n} \quad (\text{F.27})$$

for some $\hat{P}''' \in \mathcal{P}(n)$, one can see that

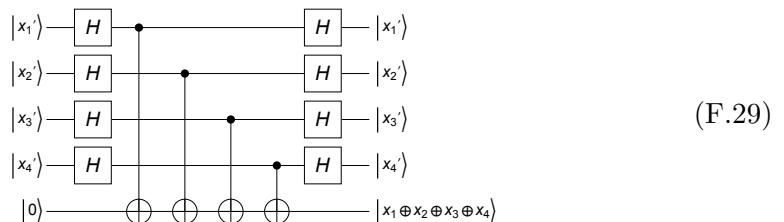
$$\hat{P}\hat{V}|y\rangle = \sum_x |x\rangle (\langle 0| \otimes \langle x|)(\hat{Z} \otimes \hat{W})\hat{U}(|0\rangle \otimes \hat{P}''|y\rangle).$$

It implies that for any $\hat{P} \in \mathcal{P}(n)$, there exist $\hat{P}''', \hat{P}''' \in \mathcal{P}(n)$ such that

$$\hat{P}''' \hat{P}\hat{V} = \hat{V}\hat{P}'''. \quad (\text{F.28})$$

As \hat{P}''' is invertible, $\hat{P}''' \hat{P}$ covers whole $\mathcal{P}(n)$, and Eq. (F.28) proves the statement.

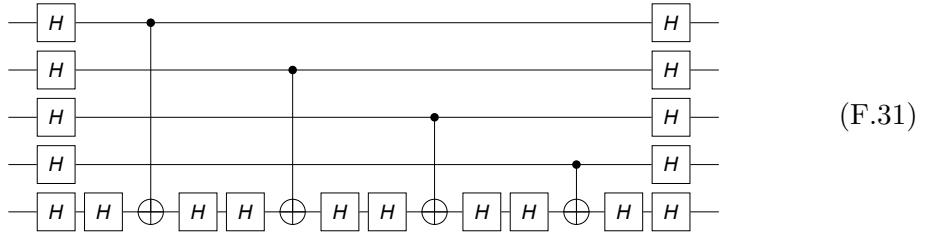
Problem 6.9 We start with the quantum circuit model



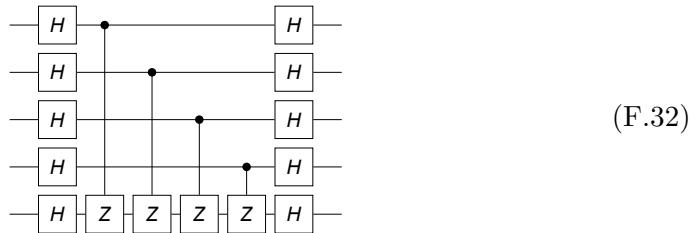
where the states $|x'\rangle$ with $x = 0, 1$ are defined by

$$|0'\rangle \equiv |+\rangle := \frac{|0\rangle + |1\rangle}{\sqrt{2}}, \quad |1'\rangle \equiv |-\rangle := \frac{|0\rangle - |1\rangle}{\sqrt{2}}. \quad (\text{F.30})$$

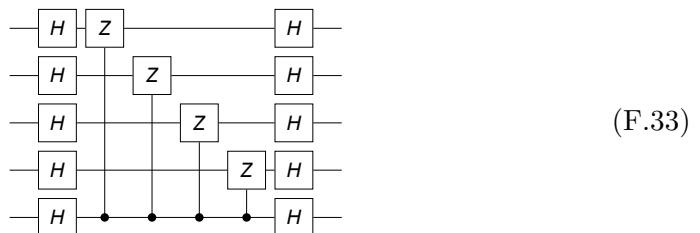
As $\hat{H}^2 = \hat{I}$, it is equivalent to the following quantum circuit model



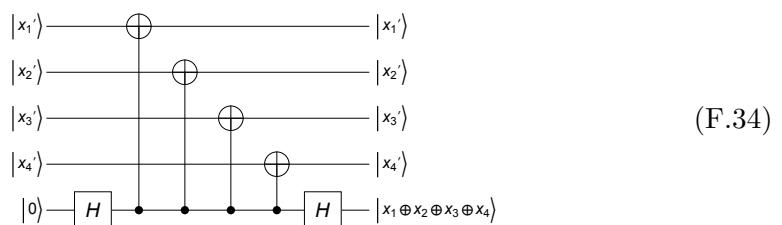
Now we use the identity in Eq. (2.43) to get



Since CZ gates are symmetric about control and target qubits, the above quantum circuit model is equivalent to the following



Finally, we use the identity in Eq. (2.43) again to arrive at the following quantum circuit model



Bibliography

Aaronson, S. & D. Gottesman, Physical Review A **70** (5) (2004). “Improved simulation of stabilizer circuits”. DOI: [10.1103/physreva.70.052328](https://doi.org/10.1103/physreva.70.052328) arXiv:[quant-ph/0406196](https://arxiv.org/abs/quant-ph/0406196)

Aharonov, Y. & J. Anandan, Phys. Rev. Lett. **58** (16), 1593 (1987). “Phase Change During a Cyclic Quantum Evolution”. DOI: [10.1103/PhysRevLett.58.1593](https://doi.org/10.1103/PhysRevLett.58.1593)

Alicea, J., Y. Oreg, G. Refael, F. von Oppen, & M. P. A. Fisher, Nat Phys **7** (5), 412 (2011). “Non-Abelian statistics and topological quantum information processing in 1D wire networks”. DOI: [10.1038/nphys1915](https://doi.org/10.1038/nphys1915)

Anandan, J., Physics Letters A **133** (4-5), 171 (1988). “Non-adiabatic non-abelian geometric phase”. DOI: [10.1016/0375-9601\(88\)91010-9](https://doi.org/10.1016/0375-9601(88)91010-9)

Aspect, A., P. Grangier, & G. Roger, Phys. Rev. Lett. **47** (7), 460 (1981). “Experimental Tests of Realistic Local Theories via Bell’s Theorem”.

Barenco, A., C. H. Bennett, R. Cleve, *et al.*, Physical Review A **52** (5), 3457 (1995). “Elementary gates for quantum computation”. DOI: [10.1103/physreva.52.3457](https://doi.org/10.1103/physreva.52.3457) arXiv:[quant-ph/9503016](https://arxiv.org/abs/quant-ph/9503016)

Bell, J. S., Rev. Mod. Phys. **38** (3), 447 (1966). “On the Problem of Hidden Variables in Quantum Mechanics”.

Bennett, C. H., D. P. DiVincenzo, J. A. Smolin, & W. K. Wootters, Phys. Rev. A **54** (5), 3824 (1996). “Mixed-state entanglement and quantum error correction”. DOI: [10.1103/PhysRevA.54.3824](https://doi.org/10.1103/PhysRevA.54.3824) arXiv:[quant-ph/9604024](https://arxiv.org/abs/quant-ph/9604024)

Bennett, C. H. & S. J. Wiesner, Phys. Rev. Lett. **69** (20), 2881 (1992). “Communication via one- and two-particle operators on Einstein-Podolsky-Rosen states”.

Bergou, J. A., U. Herzog, & M. Hillery, “Discrimination of Quantum States,” in Paris & Rehacek (2004), Chap. 11, pp. 417–465. DOI: [10.1007/978-3-540-44481-7_11](https://doi.org/10.1007/978-3-540-44481-7_11)

- Bernstein, E. & U. Vazirani, “Quantum Complexity Theory,” in *Proceedings of the 25th Annual ACM Symposium on the Theory of Computing* (ACM Press, New York, 1993), pp. 11–20.
- Bernstein, E. & U. Vazirani, SIAM Journal on Computing **26** (5), 1411 (1997). “Quantum Complexity Theory”. DOI: [10.1137/s0097539796300921](https://doi.org/10.1137/s0097539796300921)
- Berry, M. V., Proc. R. Soc. London A **392**, 45 (1984). “Quantal Phase Factors Accompanying Adiabatic Changes”.
- Blum, K., *Density Matrix Theory and Applications*, Vol. 64 of *Springer Series on Atomic, Optical, and Plasma Physics* (Springer Berlin Heidelberg, 2012), 3rd ed., ISBN 978-3-642-20560-6.
- Born, M., Z. Phys. **37** (12), 863 (1926). “Zur Quantenmechanik der Stoßvorgänge”.
- Bouwmeester, D., J.-W. Pan, M. Daniell, H. Weinfurter, & A. Zeilinger, Phys. Rev. Lett. **82** (7), 1345 (1999). “Observation of Three-Photon Greenberger-Horne-Zeilinger Entanglement”.
- Bravyi, S. B. & A. Y. Kitaev, arXiv:[quant-ph/9811052](https://arxiv.org/abs/quant-ph/9811052) (1998). “Quantum codes on a lattice with boundary”.
- Breuer, H.-P. & F. Petruccione, *The Theory of Open Quantum Systems* (Oxford University Press, New York, 2002).
- Browne, D. & H. Briegel, “One-Way Quantum Computation,” in *Quantum Information: From Foundations to Quantum Technology Applications*, edited by Bruß, D. & G. Leuchs (Wiley, 2016), pp. 449–473, 2nd ed. DOI: [10.1002/9783527805785.ch21](https://doi.org/10.1002/9783527805785.ch21) arXiv:[quant-ph/0603226](https://arxiv.org/abs/quant-ph/0603226)
- Calderbank, A. R. & P. W. Shor, Phys. Rev. A **54** (2), 1098 (1996). “Good quantum error-correcting codes exist”.
- Caves, C. M., Phys. Rev. D **23** (8), 1693 (1981). “Quantum-mechanical noise in an interferometer”.
- Chefles, A., “Quantum States: Discrimination and Classical Information Transmission. A Review of Experimental Progress,” in [Paris & Rehacek \(2004\)](#), Chap. 12, pp. 467–511. DOI: [10.1007/978-3-540-44481-7_12](https://doi.org/10.1007/978-3-540-44481-7_12)
- Chiaverini, J., Science **308** (5724), 997 (2005). “Implementation of the Semiclassical Quantum Fourier Transform in a Scalable System”. DOI: [10.1126/science.1110335](https://doi.org/10.1126/science.1110335)
- Choi, M.-S., J. Phys.: Condens. Matt. **15** (46), 7823 (2003). “Geometric Quantum Computation in Solid-State Qubits”. arXiv:[quant-ph/0111019](https://arxiv.org/abs/quant-ph/0111019)

- Clauser, J. F., M. A. Horne, A. Shimony, & R. A. Holt, Phys. Rev. Lett. **23**, 880 (1969). “Proposed Experiment to Test Local Hidden-Variable Theories”.
- Cleve, R., A. Ekert, C. Macchiavello, & M. Mosca, Proceedings of the Royal Society A **454 (1969)**, 339 (1998). “Quantum algorithms revisited”. DOI: [10.1098/rspa.1998.0164](https://doi.org/10.1098/rspa.1998.0164) arXiv:[quant-ph/9708016](https://arxiv.org/abs/quant-ph/9708016)
- Cleve, R. & D. Gottesman, Physical Review A **56 (1)**, 76 (1997). “Efficient computations of encodings for quantum error correction”. DOI: [10.1103/physreva.56.76](https://doi.org/10.1103/physreva.56.76) arXiv:[quant-ph/9607030](https://arxiv.org/abs/quant-ph/9607030)
- Cornwell, J. F., *Group Theory in Physics*, Vol. I (Academic Press, Orlando, 1984).
- Cornwell, J. F., *Group Theory in Physics: An Introduction* (Academic Press, San Diego, 1997).
- Das, A., Y. Ronen, Y. Most, Y. Oreg, M. Heiblum, & H. Shtrikman, Nat Phys **8 (12)**, 887 (2012). “Zero-bias peaks and splitting in an Al-InAs nanowire topological superconductor as a signature of Majorana fermions”.
- Deng, M. T., C. L. Yu, G. Y. Huang, M. Larsson, P. Caroff, & H. Q. Xu, Nano Letters **12 (12)**, 6414 (2012). “Anomalous Zero-Bias Conductance Peak in a Nb–InSb Nanowire–Nb Hybrid Device”.
- Dennis, E., A. Kitaev, A. Landahl, & J. Preskill, Journal of Mathematical Physics **43 (9)**, 4452 (2002). “Topological quantum memory”. DOI: [10.1063/1.1499754](https://doi.org/10.1063/1.1499754) arXiv:[quant-ph/0110143](https://arxiv.org/abs/quant-ph/0110143)
- Deutsch, D., Proc. R. Soc. London A **400**, 97 (1985). “Quantum theory, the Church-Turing principle and the universal quantum computer”. DOI: [10.1098/rspa.1985.0070](https://doi.org/10.1098/rspa.1985.0070)
- Deutsch, D. & R. Jozsa, Proceedings of the Royal Society A: Mathematical, Physical and Engineering Sciences **439 (1907)**, 553 (1992). “Rapid Solution of Problems by Quantum Computation”. DOI: [10.1098/rspa.1992.0167](https://doi.org/10.1098/rspa.1992.0167)
- DiVincenzo, D. P., Fortschr. Phys. **48**, 771 (2000). “The Physical Implementation of Quantum Computation”. DOI: [10.1002/1521-3978\(200009\)48:9/11<771::AID-PROP771>3.0.CO;2-E](https://doi.org/10.1002/1521-3978(200009)48:9/11<771::AID-PROP771>3.0.CO;2-E) arXiv:[quant-ph/0002077](https://arxiv.org/abs/quant-ph/0002077)
- Dum, R., A. S. Parkins, P. Zoller, & C. W. Gardiner, Phys. Rev. A **46 (7)**, 4382 (1992). “Monte Carlo simulation of master equations in quantum optics for vacuum, thermal, and squeezed reservoirs”.
- Einstein, A., B. Podolsky, & N. Rosen, Phys. Rev. **47**, 777 (1935). “Can Quantum-Mechanical Description of Physical Reality Be Considered Complete?”

- Fowler, A. G., M. Mariantoni, J. M. Martinis, & A. N. Cleland, Physical Review A **86** (3), 032324 (2012). “Surface codes: Towards practical large-scale quantum computation”. DOI: [10.1103/physreva.86.032324](https://doi.org/10.1103/physreva.86.032324) arXiv:[1208.0928](https://arxiv.org/abs/1208.0928)
- Freedman, M. H., Foundations of Computational Mathematics **1** (2), 183 (2001). “Quantum Computation and the Localization of Modular Functors”. DOI: [10.1007/s102080010006](https://doi.org/10.1007/s102080010006)
- Giovannetti, V., S. Lloyd, & L. Maccone, Physical Review Letters **96** (1), 010401 (2006). “Quantum Metrology”. DOI: [10.1103/PhysRevLett.96.010401](https://doi.org/10.1103/PhysRevLett.96.010401) arXiv:[quant-ph/0509179](https://arxiv.org/abs/quant-ph/0509179)
- Goldstein, S., Phys. Rev. Lett. **72** (13), 1951 (1994). “Nonlocality without inequalities for almost all entangled states for two particles”.
- Gottesman, D., Physical Review A **54** (3), 1862 (1996). “Class of quantum error-correcting codes saturating the quantum Hamming bound”. DOI: [10.1103/physreva.54.1862](https://doi.org/10.1103/physreva.54.1862) arXiv:[quant-ph/9604038](https://arxiv.org/abs/quant-ph/9604038)
- Gottesman, D., *Stabilizer Codes and Quantum Error Correction*, Ph.D. thesis, California Institute of Technology, Pasadena, California (1997). arXiv:[quant-ph/9705052](https://arxiv.org/abs/quant-ph/9705052)
- Gottesman, D., Phys. Rev. A **57** (1), 127 (1998). “Theory of fault-tolerant quantum computation”. DOI: [10.1103/PhysRevA.57.127](https://doi.org/10.1103/PhysRevA.57.127) arXiv:[quant-ph/9702029](https://arxiv.org/abs/quant-ph/9702029)
- Gottesman, D., “The Heisenberg Representation of Quantum Computers,” in *Group22 : proceedings of XXII International Colloquium on Group Theoretical Methods in Physics : Hobart, July 13-17, 1998*, edited by Corney, S. P., R. Delbourgo, & P. D. Jarvis (International Press, Cambridge, MA, 1999), ISBN 978-1571460547. arXiv:[quant-ph/9807006](https://arxiv.org/abs/quant-ph/9807006)
- Greenberger, D. M., M. A. Horne, A. Shimony, & A. Zeilinger, Ame. J. Phys. **58**, 1131 (1990). “Bell’s theorem without inequalities”. DOI: [10.1119/1.16243](https://doi.org/10.1119/1.16243)
- Greenberger, D. M., M. A. Horne, & A. Zeilinger, “Going beyond Bell’s theorem,” in *Bell’s Theorem, Quantum Theory, and Conceptions of the Universe*, edited by Kafatos, M. (Kluwer Academic, Dordrecht, The Netherlands, 1989). arXiv:[0712.0921](https://arxiv.org/abs/0712.0921)
- Griffiths, R. B. & C.-S. Niu, Physical Review Letters **76** (17), 3228 (1996). “Semiclassical Fourier Transform for Quantum Computation”. DOI: [10.1103/physrevlett.76.3228](https://doi.org/10.1103/physrevlett.76.3228) arXiv:[quant-ph/9511007](https://arxiv.org/abs/quant-ph/9511007)
- Grover, L. K., “A fast quantum mechanical algorithm for database search,” in *Proceedings of the 28th Annual ACM Symposium on the Theory of Computing* (ACM Press, New York, 1996), p. 212. arXiv:[quant-ph/9605043](https://arxiv.org/abs/quant-ph/9605043)

- Grover, L. K., Phys. Rev. Lett. **79** (2), 325 (1997). “Quantum Mechanics Helps in Searching for a Needle in a Haystack”.
- Hardy, L., Phys. Rev. Lett. **68** (20), 2981 (1992). “Quantum Mechanics, Local Realistic Theories, and Lorentz-Invariant Realistic Theories”.
- Hardy, L., Phys. Rev. Lett. **71**, 1665 (1993). “Nonlocality for two particles without inequalities for almost all entangled states”.
- Higgins, B. L., D. W. Berry, S. D. Bartlett, H. M. Wiseman, & G. J. Pryde, Nature **450** (7168), 393 (2007). “Entanglement-free Heisenberg-limited phase estimation”. DOI: [10.1038/nature06257](https://doi.org/10.1038/nature06257) arXiv:[0709.2996](https://arxiv.org/abs/0709.2996)
- Horodecki, M., P. Horodecki, & R. Horodecki, Phys. Lett. A **223** (1), 1 (1996). “Separability of mixed states: necessary and sufficient conditions”. DOI: [10.1016/0375-9601\(95\)00930-2](https://doi.org/10.1016/0375-9601(95)00930-2)
- Jiang, M., S. Luo, & S. Fu, Physical Review A **87** (2) (2013). “Channel-state duality”. DOI: [10.1103/physreva.87.022310](https://doi.org/10.1103/physreva.87.022310)
- Kitaev, A. Y., Electronic Colloquium on Computational Complexity **3**, 3 (1996). “Quantum measurements and the Abelian Stabilizer Problem”. arXiv:[quant-ph/9511026](https://arxiv.org/abs/quant-ph/9511026)
- Kitaev, A. Y., Russian Mathematical Surveys **52** (6), 1191 (1997). “Quantum computations: algorithms and error correction”.
- Kitaev, A. Y., Physics-Uspekhi **44** (10S), 131 (2001). “Unpaired Majorana fermions in quantum wires”. DOI: [10.1070/1063-7869/44/10S/S29](https://doi.org/10.1070/1063-7869/44/10S/S29) arXiv:[cond-mat/0010440](https://arxiv.org/abs/cond-mat/0010440)
- Kitaev, A. Y., Ann. Phys. **303** (1), 2 (2003). “Fault-tolerant quantum computation by anyons”. DOI: [10.1016/S0003-4916\(02\)00018-0](https://doi.org/10.1016/S0003-4916(02)00018-0) arXiv:[quant-ph/9707021](https://arxiv.org/abs/quant-ph/9707021)
- Laflamme, R., C. Miquel, J. P. Paz, & W. H. Zurek, Physical Review Letters **77** (1), 198 (1996). “Perfect Quantum Error Correcting Code”. DOI: [10.1103/physrevlett.77.198](https://doi.org/10.1103/physrevlett.77.198) arXiv:[quant-ph/9602019](https://arxiv.org/abs/quant-ph/9602019)
- Lang, S., *Introduction to Linear Algebra*, Undergraduate Texts in Mathematics (Springer New York, New York, 1986), 2nd ed., ISBN 9781461210702. DOI: [10.1007/978-1-4612-1070-2](https://doi.org/10.1007/978-1-4612-1070-2)
- Lang, S., *Linear Algebra* (Springer, Berlin, 1987), 3rd ed., ISBN 978-1-4757-1949-9. DOI: [10.1007/978-1-4757-1949-9](https://doi.org/10.1007/978-1-4757-1949-9)
- Loss, D. & D. P. DiVincenzo, Phys. Rev. A **57** (1), 120 (1998). “Quantum comutation with quantum dots”.

- Lundeen, J. S., B. Sutherland, A. Patel, C. Stewart, & C. Bamber, Nature **474** (7350), 188 (2011). “Direct measurement of the quantum wavefunction”. DOI: [10.1038/nature10120](https://doi.org/10.1038/nature10120)
- Mourik, V., K. Zuo, S. M. Frolov, S. R. Plissard, E. P. A. M. Bakkers, & L. P. Kouwenhoven, Science **336** (6084), 1003 (2012). “Signatures of Majorana Fermions in Hybrid Superconductor-Semiconductor Nanowire Devices”.
- Nadj-Perge, S., I. K. Drozdov, J. Li, *et al.*, Science **346** (6209), 602 (2014). “Observation of Majorana fermions in ferromagnetic atomic chains on a superconductor”. DOI: [10.1126/science.1259327](https://doi.org/10.1126/science.1259327) arXiv:<http://www.sciencemag.org/content/346/6209/602.full.pdf>
- Nakazato, H., Y. Hida, K. Yuasa, B. Militello, A. Napoli, & A. Messina, Physical Review A **74** (6), 062113 (2006). “Solution of the Lindblad equation in the Kraus representation”. DOI: [10.1103/physreva.74.062113](https://doi.org/10.1103/physreva.74.062113) arXiv:[quant-ph/0606193](https://arxiv.org/abs/quant-ph/0606193)
- Nielsen, M. & I. L. Chuang, *Quantum computation and quantum information* (Cambridge University Press, New York, 2011), 10th anniversary ed., ISBN 978-1107002173.
- Pan, J.-W., D. Bouwmeester, M. Daniell, H. Weinfurter, & A. Zeilinger, Nature **403**, 515 (2000). “Experimental test of quantum nonlocality in three-photon Greenberger-Horne-Zeilinger entanglement”.
- Paris, M. & J. Rehacek, eds., *Quantum State Estimation*, Vol. 649 of *Lecture Notes in Physics* (Springer Berlin Heidelberg, Berlin, 2004), ISBN 9783540444817. DOI: [10.1007/b98673](https://doi.org/10.1007/b98673)
- Peres, A., Phys. Rev. Lett. **77** (8), 1413 (1996). “Separability Criterion for Density Matrices”. DOI: [10.1103/PhysRevLett.77.1413](https://doi.org/10.1103/PhysRevLett.77.1413) arXiv:[quant-ph/9604005](https://arxiv.org/abs/quant-ph/9604005)
- Plenio, M. B. & P. L. Knight, Rev. Mod. Phys. **70** (1), 101 (1998). “The quantum-jump approach to dissipative dynamics in quantum optics”.
- Raussendorf, R. & H. J. Briegel, Phys. Rev. Lett. **86** (22), 5188 (2001). “A One-Way Quantum Computer”.
- Raussendorf, R., D. Browne, & H. Briegel, Journal of Modern Optics **49** (8), 1299 (2002). “The one-way quantum computer—a non-network model of quantum computation”. DOI: [10.1080/09500340110107487](https://doi.org/10.1080/09500340110107487) arXiv:[quant-ph/0108118](https://arxiv.org/abs/quant-ph/0108118)
- Raussendorf, R., D. E. Browne, & H. J. Briegel, Phys. Rev. A **68** (2), 022312 (2003). “Measurement-based quantum computation on cluster states”. DOI: [10.1103/PhysRevA.68.022312](https://doi.org/10.1103/PhysRevA.68.022312) arXiv:[quant-ph/0301052](https://arxiv.org/abs/quant-ph/0301052)

- Schwinger, J., Proceedings of the National Academy of Sciences **45** (10), 1542 (1959). “The Algebra Of Microscopic Measurement”. DOI: [10.1073/pnas.45.10.1542](https://doi.org/10.1073/pnas.45.10.1542)
- Shor, P. W., “Algorithms for Quantum Computation: Discrete Logarithms and Factoring,” in *Proceedings of the 35th Annual Symposium on Foundations of Computer Science* (IEEE Computer Society, Washington, DC, USA, 1994), SFCS ’94, pp. 124–134. DOI: [10.1109/SFCS.1994.365700](https://doi.org/10.1109/SFCS.1994.365700)
- Shor, P. W., SIAM Journal on Computing **26** (5), 1484 (1997). “Polynomial-Time Algorithms for Prime Factorization and Discrete Logarithms on a Quantum Computer”. arXiv:[quant-ph/9508027](https://arxiv.org/abs/quant-ph/9508027)
- Simon, D. R., SIAM Journal on Computing **26** (5), 1474 (1997). “On the Power of Quantum Computation”. DOI: [10.1137/s0097539796298637](https://doi.org/10.1137/s0097539796298637)
- Sjöqvist, E., D. M. Tong, L. Mauritz Andersson, B. Hessmo, M. Johansson, & K. Singh, New Journal of Physics **14** (10), 103035 (2012). “Non-adiabatic holonomic quantum computation”. DOI: [10.1088/1367-2630/14/10/103035](https://doi.org/10.1088/1367-2630/14/10/103035) arXiv:[1107.5127](https://arxiv.org/abs/1107.5127)
- Smolin, J. A. & D. P. DiVincenzo, Phys. Rev. A **53** (4), 2855 (1996). “Five two-bit quantum gates are sufficient to implement the quantum Fredkin gate”. DOI: [10.1103/PhysRevA.53.2855](https://doi.org/10.1103/PhysRevA.53.2855)
- Steane, A. M., Phys. Rev. Lett. **77** (5), 793 (1996). “Error Correcting Codes in Quantum Theory”.
- Størmer, E., *Positive Linear Maps of Operator Algebras* (Springer, Berlin, 2013), ISBN 9783642343698. DOI: [10.1007/978-3-642-34369-8](https://doi.org/10.1007/978-3-642-34369-8)
- Vallone, G. & D. Dequal, Physical Review Letters **116** (4), 040502 (2016). “Strong Measurements Give a Better Direct Measurement of the Quantum Wave Function”. DOI: [10.1103/physrevlett.116.040502](https://doi.org/10.1103/physrevlett.116.040502) arXiv:[1504.06551](https://arxiv.org/abs/1504.06551)
- Vedral, V., A. Barenco, & A. Ekert, Physical Review A **54** (1), 147 (1996). “Quantum networks for elementary arithmetic operations”. DOI: [10.1103/physreva.54.147](https://doi.org/10.1103/physreva.54.147) arXiv:[quant-ph/9511018](https://arxiv.org/abs/quant-ph/9511018)
- Wang, C., J. Harrington, & J. Preskill, Annals of Physics **303** (1), 31 (2003). “Confinement-Higgs transition in a disordered gauge theory and the accuracy threshold for quantum memory”. DOI: [10.1016/s0003-4916\(02\)00019-2](https://doi.org/10.1016/s0003-4916(02)00019-2)
- Weyl, H., *The theory of groups and quantum mechanics* (Dover, London, 1931).
- Wigner, E. P., *Group Theory and its Application to the Quantum Mechanics of Atomic Spectra* (Academic Press, New York, 1959), english translation ed.

- Wilczek, F. & A. Zee, Phys. Rev. Lett. **52** (24), 2111 (1984). “Appearance of Gauge Structure in Simple Dynamical Systems”. DOI: [10.1103/PhysRevLett.52.2111](https://doi.org/10.1103/PhysRevLett.52.2111)
- Wilmut, I., A. E. Schnieke, J. McWhir, A. J. Kind, & K. H. S. Campbell, Nature **385** (6619), 810 (1997). “Viable offspring derived from fetal and adult mammalian cells”.
- Wooters, W. K. & W. H. Zurek, Nature **299**, 802 (1982). “A single quantum cannot be cloned”.
- Zanardi, P. & M. Rasetti, Phys. Lett. A **264** (2-3), 94 (1999). “Holonomic quantum computation”. DOI: [10.1016/S0375-9601\(99\)00803-8](https://doi.org/10.1016/S0375-9601(99)00803-8) arXiv:[quant-ph/9904011](https://arxiv.org/abs/quant-ph/9904011)
- Zurek, W. H., Phys. Today **44** (10), 36 (1991). “Decoherence and the transition from quantum to classical”.
- Zurek, W. H., Nature **404**, 130 (2000). “Quantum cloning: Schrodinger’s sheep”. DOI: [10.1038/35004684](https://doi.org/10.1038/35004684)
- Zurek, W. H., Los Alamos Science **27**, 2 (2002). “Decoherence and the Transition from Quantum to Classical: Revisited”.

Index

- Abelian group, 252, 341
amplitude damping, 215
ancillary qubit, 78
- Bell basis, *see also* Bell states, 139
Bell measurement, 87, 139, 204
Bell states, 87, 139
Bell's inequality, 136
Bell's test, 136
Bernstein-Vazirani algorithm, 142, 150
birthday paradox, 151
bit flip, 42
bitwise AND, 75
Bloch space, 256, 257
Bloch sphere, 19
Bloch states, 181
Bloch vector, 19
bra-ket notation, 40
bright state, 108
- Calderbank-Shor-Steane codes, 277
center, 250
channel-state duality, 203
Choi isomorphism, 201–204, 329
Choi matrix, 202, 321, *see also* Choi operator
Choi operator, *see also* Choi matrix, 202, 229, 328
Choi vector, 200, 204
classical communication channel, 139
classical feedback control, *see also* semi-classical feedback control, 264
Clifford circuit, 261, 262
Clifford group, 256, 260, 294
Clifford operators, *see also* Pauli operators, 256, 274
- closed system, 10, 191, 192
closure relation, 195
cluster state, 89, 116, 127
CNOT, 51, 61, 81, 116, 139, 256
 controlled-NOT gate, 51
 multi-qubit controlled-NOT, 77
CNOT gate, 58, 85, 103, 182
code space, 232, 244
code words, 237
complementarity principle, 9
completely positive and trace-preserving supermap, 193
completely positive supermap, 192–194, 203, 325, 326
completeness relation, 32, 211, 308
computational basis, *see also* logical basis
conjugation, 256, 258
continued fraction, 177
control qubit, 51
controlled unitary gate, 167
controlled-*U* gate, 51, 189
 multi-qubit controlled-*U* gate, 64, 74
convergent of a continued fraction
 continued fraction, 178
convex linear, 193
coset, 250, 255, 338
CSS codes, *see also* Calderbank-Shor-Steane codes
cyclic evolution, 114
cyclic group, 256, 341
cyclic groups, 336
CZ, 116
 controlled-Z gate, 56
CZ gate, 56, 103–105, 129, 256

- damping operator, 213, 216, 223
 dark state, 130
 dark states, 108
 decoherence, 193, 231
 density matrix, 16
 density operator, 16, 191–193, 203, 219, 315
 dephasing, 206, 207
 depolarizing process, 216
 Deutsch-Jozsa algorithm, 142
 Deutsch-Jozsa problem, 151
 direct product group, 341
 discrete Fourier transform, 155, 156, 174
 discrete logarithm, 156
 DiVincenzo criteria, 94
 effective Hamiltonian, 212, 216, 223
 elementary quantum logic gates, 40
 Elements, 9, 178
 entangled state, 13, 15, 52
 entanglement, 21, 126, 139, 145
 entanglement fidelity, 202
 environment, 191
 error correction conditions, 270
 error syndrome, 233
 error-detection, 233
 error-recovery, 233
 Euclid, 178
 Euclid of Alexandria, 9
 Euclidean algorithm, 178
 Euler rotation, 49, 120
 Euler angles, 49
 exclusive OR, 53
 factor group, 250, 338
 fidelity, 202, 297
 flux quantization, 93
 Fredkin gate, 80, 81
 gate teleportation, 204
 gauge transformation, 115
 generalized interaction picture, 223
 generalized measurement, 210
 geometric phase, 106
 geometric quantum computation, 112, 114, 115
 golden ratio, 178
 Gottesman vector, 251, 252, 255
 Gottesman vectors, 353
 Gottesman-Knill theorem, 261
 graph state, *see also* cluster state, 116, 125–127, 262
 Gray code, 64, 75
 Gray code sequence, 75
 Greenberger-Horne-Zeilinger (GHZ) experiment
 Greenberger-Horne-Zeilinger (GHZ) state, 262
 Greenberger-Horne-Zeilinger (GHZ) state, 56
 Greenberger-Horne-Zeilinger state, 89
 group, 333
 group generators, 248, 263, 335
 group theory, 247, 333
 Grover operator, 187
 Grover rotation, 188
 Grover's algorithm, 182
 Grover's diffusion operator, 183
 Grover, L. K., 182
 Grover's algorithm, *see also* quantum search algorithm
 Hadamard gate, 44, 85, 88, 104, 117, 119, 139, 256
 Hadamard matrix, 44
 Hardy's test, 137
 Heisenberg exchange interaction, 103
 Heisenberg limit, 32
 Hermitian operator, 192, 324
 hidden subgroup problem, 156, 166, 173–175
 hidden subgroup problem, 156
 Hilbert space, 10, 94
 homomorphism, 337
 Householder reflection, *see also* Householder transformation
 Householder transformation, 183, 184, 188

- inertial force, 99
inertial frame, 98
initialization, 95
invariant subgroup, 250
inverse quantum Fourier transform
 quantum Fourier transform, 162
irreversible population loss, 213
Ising exchange interaction, 105
Ising exchange interaction, 104, 105
isomorphism, 337
- Josephson inductance, 93
- kinetic inductance, 93
- Kraus element, 263
- Kraus elements, 194, 207, 215, 229, 326
 orthogonal Kraus elements, 326
- Kraus maps, *see also* Kraus elements,
 see also Kraus elements
- Kraus operator-sum representation, *see also*
 Kraus representation
- Kraus operators, *see also* Kraus elements
- Kraus representation, 193, 199, 326
- Larmor precession, 97
- Linblad equation, 322
- Lindblad basis, 219
- Lindblad equation, 212, 218, 222
- Lindblad generator, 212, 213, 215, 219
- Lindblad operator, 212, 213
- linear independent, 297
- linearly dependent, 297
- logical basis, 11, 255
- logical operators, 254–256, 262–265, 273
- Markov approximation, 212
- Markov assumption, 214
- maximally entangled, 89
- maximally entangled state, 202
- measurement, 39, 95, 193
- measurement operators, 32, 211
- measurement-based quantum computation, 116
- mixed state, 15, 205
- modular exponentiation, 173, 180, 182
- modular multiplication, 168
- modular multiplication, 181, 182
- momentum basis, 172
- Newton’s laws of motion, 9
- no-cloning theorem, 28, 138, 231
- non-Abelian gauge potential, 115
- non-cloning theorem, 54
- non-Hermitian Hamiltonian, 213, 223
- non-inertial effect, 99, 114
- non-negative operator, *see also* positive
 semidefinite operator
- non-selective measurement, 211
- nonlocality, 135
- normal operator, 307
- normalizer, 256
- NOT gate, 103
- octant phase gate, 50, 85, 262
- one-way quantum computation, *see also*
 measurement-based quantum computation
- open quantum system, 191, 192
- operation time, 97
- operator-sum representation, 203, 323,
 325
- oracle, 185
- order-finding algorithm, 180
 order-finding problem, 156, 168, 180
- orthonormal basis, 308
- parallel transport, 115
- partial trace, 194
- path ordering, 115
- Pauli gates, *see also* Pauli operators
- Pauli group, 247, 252, 256, 270, 334, 340,
 341
- Pauli operator, 130, 248
- Pauli operators, 40, 88, 95, 139, 256
 Pauli X, 40
 Pauli Y, 42
 Pauli Z, 41
- period-finding algorithm, 174, 180, 181
- phase damping, 207, 215

- phase flip, 42
 phase gate, 88
 principle of deferred measurement, 163
 planar codes, 281, 287
 planar exchange interaction, *see also* XY exchange interaction
 plaquette defect, 291, 292
 plaquette operators, 281, 282, 287
 point group, 256
 position basis, 172
 positive definite operator, *see also* positive operator
 positive operator, 192, 203, 303, 307, 308, 324
 positive semidefinite operator, 195, 303, 307, 308
 postulates of quantum mechanics, 9
 POVM, 33
 POVM elements
 POVM, 33
 product group, 341
 projection operator, 203, 328
 projective measurement, 32
 pure state, 29
- quadrant phase gate, 85, 256
 quadrant phase grate, 50, 257
 quantum entanglement, 15
 quantum channel, 193
 quantum circuit model
 quantum circuit diagram, 39
 quantum communication, 28
 quantum computer, 93
 quantum computer architecture, 93
 quantum decoherence, *see also* decoherence
 quantum efficiency, 95
 quantum entanglement, 32, 56
 quantum entangler circuit, 52, 87
 quantum error-correction conditions, 242, 243
 quantum factorization algorithm, 133, 151, 156, 166, 180, 182
 quantum Fourier transform, 157, 167–169, 172, 173
 quantum gate teleportation, 204, 229
 quantum information theory, 323
 quantum jump approach, 213, 222
 quantum jump operator, *see also* Lindblad operator, 212, 216
 quantum logic gate, 51, 93
 quantum logic gate operation, 39
 quantum Markovian dynamics, 212
 quantum master equation, *see also* Lindblad equation, 218
 quantum non-demolition measurement, 127
 quantum operation, 192, 203, 211, 229
 quantum operation formalism, 27
 quantum oracle, 142, 143, 148, 152, 174, 175, 188, 189
 quantum parallelism, 142
 quantum phase estimation, 63, 89, 156, 166, 167, 173, 175, 181
 quantum register, 45, 53
 quantum search algorithm, 182
 quantum state, 10
 quantum statistical mechanics, 323
 quantum teleportation, 15, 56, 134, 138, 204
 qubit, 39, 94
 quantum bit, 39
 quantum phase estimation, 166
 Rabi oscillation, 99, 114
 Rabi frequency, 99
 reduced density matrix, 89
 reference space, 202
 resonance, 99
 rotating-wave approximation, 102
 rotating frame, 98
 Hamiltonian in the rotating frame, 99
 time-evolution operator in the rotating frame, 99
 rotation operator, 48, 97

- scalable system, 94
Schmidt decomposition, 13
selective measurement, 211
semiclassical feedback control, *see also* classical feedback control, 264
separable state, 13
Shor's algorithm, *see also* quantum factorization algorithm, 133
Shor, Peter W., 133
Simon's algorithm, 142, 151
 Simon's problem, 151
special theory of relativity, 136
spectral decomposition, 195, 203, 308
spin-boson model, 128
stabilizer, 245, 262, 263, 293
stabilizer circuit, 261, 262
stabilizer codes, 244, 253, 277
stabilizer formalism, 126, 244
stabilizer subgroup, *see also* stabilizer,
 see also stabilizer
standard quantum limit, 32
state vector, 10
statistical ensemble, 15
statistical mixture, 315
super-mapping, *see also* supermap
superdense coding, 134
supermap, 192, 203, 321, 323
superoperator, 193, 212, 315, 321
surface codes, 281
SWAP, 58, 81
SWAP gate, 58, 103, 129, 134
 $\sqrt{\text{SWAP}}$ gate, 60, 103, 104, 129
 SWAP gate, 103

target qubit, 51
tensor-product basis, 13
tensor-product space, 13
time ordering, 114
Toffoli gate, 79, 81, 85, 262
topological quantum computation, 115
toric code, 281
toric codes, 281
trace, 193
trace Hermitian product, 194, 199, 319
trace product, *see also* trace Hermitian product
translational freedom of Lindblad operators, 213
two-level unitary transformation, 83
two-level unitary transformation, 67, 69, 77
 two-level unitary matrix, 72, 74

uncertainty principle, 28
unconditional security, 28
unitary freedom of Kraus elements, 200, 213
unitary freedom of Lindblad operators, 213, 215
unitary group, 40
unitary matrix, 40
unitary representation, 193
universal quantum computation, 40, 67, 74, 85
 universal set of quantum gate operations, 95
 universal set of quantum logic gates, 82
universal set of classical logic gates, 82
unstructure search, 185
unstructured search, 182, 185

vector space of linear maps, 318
vector space of linear operators, 16
vertex defect, 291, 292
vertex operator, 292
vertex operators, 281, 287
von Neumann entropy, 22
von Neumann scheme of measurement, 171, 172

wave-particle duality, 9

XOR, *see also* exclusive OR
XY exchange interaction, 104