

Introduction to Quantum Information Science

[Artur Ekert](#), [Timothy Hosgood](#), [Alastair Kay](#), [Chiara Macchiavello](#)

Last updated: 11 April 2024

Contents

Introduction	7
Plan and intended audience	7
Notes on this PDF	8
How to cite this book	8
Acknowledgements	9
Some mathematical preliminaries	10
0.1 Complex numbers	10
0.2 Euclidean vectors and vector spaces	13
0.3 Bras and kets	15
0.4 Daggers	17
0.5 Geometry	17
0.6 Operators	18
0.7 Eigenvalues and eigenvectors	19
0.8 Outer products	21
0.9 The trace	22
0.10 Some useful identities	23
0.11 Probabilities	23
I Foundations	29
1 Quantum interference	30
1.1 Two basic rules	30
1.2 The failure of probability theory	31
1.3 The double-slit experiment	31
1.4 Superpositions	33
1.5 Interferometers	33
1.6 Qubits, gates, and circuits	36
1.7 Quantum decoherence	37
1.8 Types of computation	39
1.9 Computational complexity	41
1.10 Outlook	42
1.11 <i>Remarks and exercises</i>	42
1.11.1 A historical remark	42
1.11.2 Modifying the Born rule	43
1.11.3 Many computational paths	43
1.11.4 Distant photon emitters	44
1.11.5 Quantum Turing machines	44
1.11.6 More time, more memory	44
1.11.7 Asymptotic behaviour: big-O	44
1.11.8 Polynomial is good, and exponential is bad	45

1.11.9	Imperfect prime tester	45
1.11.10	Imperfect decision maker	45
2	Qubits	47
2.1	Composing quantum operations	47
2.2	Quantum bits, called “qubits”	48
2.3	Quantum gates and circuits	49
2.4	Single qubit interference	50
2.5	The square root of NOT	52
2.6	Phase gates galore	53
2.7	Pauli operators	54
2.8	From bit-flips to phase-flips, and back again	55
2.9	Any unitary operation on a single qubit	55
2.10	The Bloch sphere	56
2.11	Drawing points on the Bloch sphere	58
2.12	Composition of rotations	58
2.13	A finite set of universal gates	59
2.14	<i>Remarks and exercises</i>	59
2.14.1	One simple circuit	59
2.14.2	Change of basis	59
2.14.3	Operators as unitary matrices	60
2.14.4	Completing an orthonormal basis	60
2.14.5	Some sums of inner products	60
2.14.6	Some circuit identities	60
2.14.7	Unitaries preserve length	61
2.14.8	Unknown phase	61
2.14.9	One of the many cross-product identities	61
3	Quantum gates	63
3.1	Beam-splitters: physics against logic	63
3.2	Beam-splitters: quantum interference, revisited	66
3.3	The Pauli matrices, algebraically	69
3.4	Unitaries as rotations	70
3.5	Universality, again	74
3.6	Some quantum dynamics	75
3.7	<i>Remarks and exercises</i>	79
3.7.1	Quantum bomb tester	79
3.7.2	Orthonormal Pauli basis	80
3.7.3	Pauli matrix expansion coefficients	80
3.7.4	Linear algebra of the Pauli vector	81
3.7.5	Matrix Euler formula	81
3.7.6	Special orthogonal matrix calculations	81
3.7.7	Phase as rotation	82
3.7.8	Calculating a Pauli rotation	82
3.7.9	Geometry of the Hadamard	82
3.7.10	Swiss Granite Fountain	82
3.7.11	Dynamics in a magnetic field	83
4	Measurements	84
4.1	Hilbert spaces, briefly	84
4.2	Complete measurements	84
4.3	The projection rule, and incomplete measurements	86
4.4	Example of an incomplete measurement	87
4.5	Observables	88
4.6	Compatible observables and the uncertainty relation	89
4.7	Quantum communication	92
4.8	Basic quantum coding and decoding	93

4.9	Distinguishing non-orthogonal states	94
4.10	Wiesner's quantum money	96
4.11	Quantum theory, formally	96
4.12	<i>Remarks and exercises</i>	99
4.12.1	Projector?	99
4.12.2	Knowing the unknown	99
4.12.3	Measurement and idempotents	99
4.12.4	Unitary transformations of measurements	99
4.12.5	Optimal measurement	100
4.12.6	Alice knows what Bob did	100
4.12.7	The Zeno effect	100
5	Entanglement	101
5.1	A very brief history	101
5.2	From one qubit to two	101
5.3	Quantum theory, formally (continued)	102
	Tensor products	102
5.4	More qubits, and binary representations	104
5.5	Separable or entangled?	105
5.6	Controlled-NOT	107
5.7	Bell states	108
5.8	Quantum teleportation	109
5.9	No-cloning, and other no-go theorems	111
5.10	Controlled-phase and controlled-U	113
5.11	Universality, revisited	115
5.12	Phase kick-back	115
5.13	Density operators, and other things to come	117
5.14	<i>Remarks and exercises</i>	117
5.14.1	Why qubits, subsystems, and entanglement?	117
5.14.2	Entangled or not?	118
5.14.3	Instantaneous communication	118
5.14.4	SWAP circuit	119
5.14.5	Controlled-NOT circuit	119
5.14.6	Measuring with controlled-NOT	119
5.14.7	Arbitrary controlled-U on two qubits	120
5.14.8	Entangled qubits	120
5.14.9	Quantum dense coding	120
5.14.10	Playing with conditional unitaries	121
5.14.11	Tensor products in components	121
5.14.12	Hadamard transforms in components	123
5.14.13	The Schmidt decomposition	124
II	Further foundations	126
6	Bell's theorem	127
6.1	Hidden variables	127
6.2	Quantum correlations	128
6.3	The CHSH inequality	129
6.4	Bell's theorem via CHSH	130
6.5	Tsirelson's inequality	131
6.6	Quantum randomness	132
6.7	Loopholes in Bell tests	134
6.8	<i>Remarks and exercises</i>	134
6.8.1	XOR games	134
6.8.2	XOR games for quantum key distribution	135

6.8.3	XOR games for randomness expansion	135
6.8.4	Prescribed binary randomness	136
6.8.5	Unbiasing bias	136
6.8.6	Proving Tsirelson's inequality	136
6.8.7	Detector loophole	136
6.8.8	Locality loophole	136
6.8.9	Free-will loophole	137
7	Stabilisers	138
7.1	Pauli groups	138
7.2	Pauli stabilisers	140
7.3	Single stabiliser states	144
7.4	Measuring Pauli stabilisers	145
7.5	Normal subgroups	147
7.6	Pauli normalisers	148
7.7	Clifford walks on stabiliser states	150
7.8	<i>Remarks and exercises</i>	152
7.8.1	Measuring parity	152
7.8.2	The Pauli group of three qubits	154
7.8.3	Half commuting	156
7.8.4	One out of four stabilisers	156
7.8.5	Stabilisers and projectors	156
7.8.6	Abelian Pauli quotients	157
7.8.7	Equivalent projective measurements	157
8	Density matrices	158
8.1	Definitions	158
8.2	Statistical mixtures	159
8.3	Instructive examples	161
8.4	The Bloch ball	162
8.5	Subsystems of entangled systems	164
8.6	Mixtures and subsystems	165
8.7	Partial trace, revisited	167
8.8	<i>Remarks and exercises</i>	169
8.8.1	Some density operator calculations	169
8.8.2	Purification of mixed states	169
8.8.3	Pure partial trace	169
8.8.4	Maximally Bell	169
8.8.5	Spectral decompositions and common eigenbases	169
9	Quantum channels	170
9.1	Everything is (secretly) unitary	170
9.2	Random unitaries	171
9.3	Random isometries	173
9.4	Evolution of open systems	174
9.5	Stinespring's dilation and Kraus's ambiguity	176
9.6	Single-qubit channels	179
9.7	Composition of quantum channels	180
9.8	Completely positive trace-preserving maps	182
9.9	Channel-state duality	184
9.10	The mathematics of "can" and "cannot"	189
9.11	Kraus operators, revisited	190
9.12	<i>Remarks and exercises</i>	192
9.12.1	Purifications and isometries	192
9.12.2	The Markov approximation	192
9.12.3	What use are positive maps?	193
9.12.4	Partial inner product	194

9.12.5	The “control” part of controlled-NOT	194
9.12.6	Surprisingly identical channels	195
9.12.7	Independent ancilla	196
9.12.8	Order matters?	197
9.12.9	Unchanged reduced density operator	197
9.12.10	Cooling down	197
9.12.11	No pancakes	197
9.12.12	Pauli twirl	197
9.12.13	Depolarising channel	198
9.12.14	Toffoli gate	198
9.12.15	Expressing vectors using the maximally mixed state	198
9.12.16	Complete positivity of a certain map	199
9.12.17	Duals	199
9.12.18	Trace, transpose, Choi	199
9.12.19	Entanglement witness	199
9.12.20	Almost Kraus decomposition	199
9.12.21	Tricks with a maximally mixed state	199
III	Applications and reality	201
10	Quantum algorithms	202
10.1	Quantum Boolean function evaluation	202
10.2	More phase kick-back	203
10.3	Oracles	204
10.4	Deutsch’s algorithm	204
10.5	The Bernstein–Vazirani algorithm	206
10.6	Grover’s search algorithm	208
10.7	Simon’s algorithm	211
10.8	Phase estimation	214
10.9	Quantum Fourier transform	218
10.10	Hidden-order determination	220
10.11	Shor’s algorithm	223
10.12	<i>Remarks and exercises</i>	226
10.12.1	RSA	226
10.12.2	More complexity classes	226
10.12.3	Implementing reflections	226
10.12.4	Grover’s optimality	226
10.12.5	Picking out a single state	227
10.12.6	Writing an integer as a power	227
11	Quantum cryptography	228
12	Approximation	229
12.1	Metrics	229
12.2	How far apart are two quantum states?	230
12.3	Fidelity	233
12.4	Approximating unitaries	233
12.5	Approximating generic unitaries is hard, but...	235
12.6	How far apart are two probability distributions?	237
12.7	Dealing with density operators	239
12.8	Distinguishing non-orthogonal states, again	241
12.9	Approximate phase estimation	242
12.10	How accurate is accurate enough?	244
12.11	<i>Remarks and exercises</i>	244
12.11.1	Operator decompositions	244
12.11.2	More operator norms	246

12.11.3 Fidelity in a trace norm inequality	249
12.11.4 Hamming distance	250
12.11.5 Operator norm	250
12.11.6 Tolerance and precision	251
12.11.7 Statistical distance and a special event	251
12.11.8 Joint probability distributions	251
12.11.9 Distinguishability and the trace distance	251
13 Decoherence and recoherence	252
13.1 The three-qubit code	252
13.2 Towards error correction	254
13.3 Discretisation of quantum errors	256
13.4 Digitising quantum errors	257
13.5 Recoherence	258
13.6 The classical repetition code	260
13.7 Correcting bit-flips	263
13.8 Correcting phase-flips	265
13.9 Composing correctable channels	266
13.10 Correcting any single error: Shor [[9,1,3]]	268
13.11 Error-correcting assumptions	270
13.12 <i>Remarks and exercises</i>	270
13.12.1 Decoherence-free subspaces	270
13.12.2 Repetition encoding and majority voting failure	271
13.12.3 Correcting Pauli rotations with three qubits	271
13.12.4 More on Shor [[9,1,3]]	271
13.12.5 Distillation for Bell pairs	272
13.12.6 Composing quantum codes	272
14 Quantum error correction	273
14.1 The Hamming code	273
14.2 Linear codes	278
14.3 Quantum codes from classical	279
14.4 Logical operators	282
14.5 ... and error families	285
14.6 Logical operators (a different approach)	287
14.7 Error-correcting conditions	290
14.8 Code distance and thresholds	294
14.9 Encoding circuits	295
14.10 Encoding arbitrary states	297
14.11 <i>Remarks and exercises</i>	300
14.11.1 Error correcting conditions for the three-qubit code	300
14.11.2 The smallest $d = 3$ code, full stop	301
14.11.3 Hamming code encodings and decodings	302
14.11.4 Generator and parity-check matrices	302
14.11.5 A big parity check matrix	303
14.11.6 Using Tanner graphs	303
14.11.7 Five-qubit repetition code	304
14.11.8 An error in the Steane [[7, 1, 3]] code	304
14.11.9 Non-degenerate codes	304
14.11.10 The smallest $d = 3$ CSS code	304
14.11.11 CSS codes from a single matrix	305
14.11.12 Error-correcting conditions, algebraically	305
14.11.13 Steane error correction: towards fault tolerance	305
15 Fault tolerance	307
Appendix: Further topics and selected reading	308

Introduction

This section is not yet finished.

Although almost complete, this book is still a work-in-progress — a few sections are missing, but we are constantly updating and filling in the gaps! Because of this, external links to specific chapters or sections might break as things move around.

Plan and intended audience

In this series of lectures you will learn how inherently quantum phenomena, such as quantum interference and quantum entanglement, can make information processing more efficient and more secure, even in the presence of noise.

There are many introductions to quantum information science, so it seems like a good idea to start with an explanation of why this particular one exists. When learning such a subject, located somewhere in between mathematics, physics, and computer science, there are many possible approaches, with one main factor being “how far along the scale of informal to formal do I want to be?”. In these notes we take the following philosophy: it can be both interesting and fun to cover lots of ground quickly and see as much as possible on a surface level, but it’s also good to know that there is a lot of important stuff that you’ll miss by doing this. In practice, this means that we don’t worry too much about high-level mathematics. That is not to say that we do not use mathematics “properly” — in these notes you’ll find a perfectly formal treatment of e.g. quantum channels via completely positive trace-preserving maps in the language of linear algebra — but rather than putting too many footnotes with technical caveats and explanations throughout the main text, we opt to collect them all together into one big “warning” here:

The mathematics underlying quantum theory is a vast and in-depth subject, most of which we will never touch upon, some of which we will only allude to, and the rest of which we will cover only in the level of detail necessary for our overarching goal (give or take some interesting mathematical detours).

But this then poses the question of what the overarching goal of this book actually is.

This book aims to help the eager reader understand what quantum information science is all about, and for them to realise which facets of it they would like to study in more detail.

But this does not mean that our treatment is cursory! In fact, by the end of this book you will have learnt a fair bit more than what might usually be covered in a standard quantum information science course that you would find in a mathematics masters degree, for example.

The interdisciplinary nature of this topic, combined with the diverse backgrounds that different readers have, means that some may find certain chapters easy, while others find the same ones difficult — so if things seem hard to you then don’t worry, because the next chapter might feel much easier! The only real prerequisites are a working knowledge of complex numbers and vectors and matrices; some previous exposure to elementary probability theory and Dirac bra-ket notation (for example) would be helpful, but we provide crash-course introductions to some topics like these at the end of this chapter. A basic knowledge of quantum mechanics (especially in the simple context of finite dimensional state spaces, e.g. state vectors, composite systems, unitary matrices, Born rule for quantum measurements) and some ideas from classical theoretical computer science (complexity theory, algorithms) would be helpful, but is *not at all* necessary.

Of course, even if you aren't familiar with the formal mathematics of complex numbers and linear algebra, then that shouldn't stop you from reading this book if you want to. You might be surprised at how much you can black box the bits that you don't understand. The caveat stands, however, that, to *really* get to grips with this subject, at least some knowledge of maths is necessary — and this is not a bad thing!

On that note, every chapter ends with a section called “*Remarks and exercises*”. You will find the same advice in basically every single mathematical text: even just attempting to do the exercises is almost more important than reading the actual book itself. For this book, it is doubly true that *you should at least read these sections*, because they contain not just exercises but also further content including worked exercises and further fundamental expository content.

Finally, throughout this text you will find some technical asides. These are *not at all* necessary reading, but are just there to provide the exceptionally eager reader (or perhaps those with a more formal mathematical background) with some extra context, as well as some pointers towards further reading. They are usually intentionally vague and scarce in detail.

Notes on this PDF

This book is primarily an online resource: the web version contains links to external sites, embedded videos, and improved accessibility and functionality. Note that this current PDF might be an outdated version, as can be checked by comparing the “Last updated” date to the web version (which also includes a change history).

Links to websites (as opposed to sections within this document) appear in a different colour and are underlined.

The web version of this book can be found at <https://qubit.guide>.

Technical asides.

In this PDF, the technical asides mentioned in the introduction are formatted like this. One advantage of the web version is that these are hidden by default, and so don't interrupt the main text with unnecessary asides.

How to cite this book

BibLaTeX:

```
@online{qubitguide
  author = {Ekert, A and Hosgood, T and Kay, A and Macchiavello, C}
  title = {{Introduction to Quantum Information Science}}
  url = {https://qubit.guide}
  date = {2024-04-11}
}
```

BibTeX:

```
@misc{qubitguide
  author = {Ekert, A and Hosgood, T and Kay, A and Macchiavello, C}
  title = {{Introduction to Quantum Information Science}}
  howpublished = {\url{https://qubit.guide}}
  date = {2024-04-11}
}
```

Acknowledgements

We thank the following for their helpful comments and corrections, as well as for catching typos: Zhenyu Cai, Jędrzej Burkat, Maryam Khaqan, Rolando Reiner, Jan Baltussen, Linus Kelsey, Edgardo Deza, Elizabeth Ealing, L.L. Salcedo, Giuseppe Bisicchia, Tom Scruby. We also appreciate the work of Yihui Xie in developing the [Bookdown package](#) with which this e-book was built.

The creation and hosting of the online book was made possible by the [Centre for Quantum Technologies](#), the [University of Oxford Mathematical Institute](#), and the [Okinawa Institute of Science and Technology](#).



Some mathematical preliminaries

Here we quickly recall most of the fundamental mathematical results that we will rely on in the rest of this book, most importantly *linear algebra over the complex numbers*. However, we will not introduce everything from the ground up. Most notably, we will assume that the reader understands what a **matrix** is, and how it represents a **linear transformation**; some prior exposure to **complex numbers** would be helpful.

If an equation like $\text{tr}|a\rangle\langle b| = \langle b|a\rangle$ makes sense to you, and you can find the eigenvalues and eigenvectors of a matrix like

$$\begin{bmatrix} 0 & 1+i \\ \sqrt{2}e^{-i\pi/4} & 0 \end{bmatrix}$$

then you can safely skip over this section and get started directly with Chapter 1.

As a small note on notation, we generally write “ $x := y$ ” to mean “ x is defined to be (equal to) y ”, and “ $x \equiv y$ ” to mean “ x is just another name for y ”, but sometimes we simply just write “ $x = y$ ”.

0.1 Complex numbers

One of the fundamental ingredients of quantum information science (and, indeed, of quantum physics in general) is the notion of **complex numbers**. It would be disingenuous to expect that a few paragraphs would suffice to make the reader sufficiently familiar with subject, but we try our best here to give a speedy overview of the core principles, and end with some exercises that can be a helpful indicator of which things you might want to read up on elsewhere.

The “classical” way of arriving at complex numbers is as follows: start with the **natural numbers** $\mathbb{N} = \{0, 1, 2, \dots\}$, which we can add; if we want to be able to invert addition (i.e. subtract), then we end up with the **integers** $\mathbb{Z} = \{\dots, -2, -1, 0, 1, 2, \dots\}$, which we can multiply; if we want to be able to invert multiplication (i.e. divide), then we end up with the **rationals** $\mathbb{Q} = \{\frac{p}{q} \mid p, q \in \mathbb{Z}\}$. In this process of “closure under more and more binary operations”, we have passed from a **monoid**, to a **group**, to a **field**. Algebraically, then, we seem to be done: we can do all the addition and multiplication that we like, and we can invert it whenever it makes sense to do so (e.g. we can divide, as long as it’s not by 0).

But there are lots of numbers that turn up in geometry that are not rational, such as $\sqrt{2} \approx 1.414$, $\pi \approx 3.14$, and $e \approx 2.718$. To include all of these (and simultaneously make sense of things like infinite sums, and limits), we must do some **real analysis** — something which we won’t touch upon here — to end up with the **real numbers** \mathbb{R} . These form a field, just like the rationals, but now we don’t have any “gaps” in our number line. So what’s left to do?

Well the reals have one big problem: they are not **algebraically closed**. That is, there exist polynomials with no roots, i.e. equations of the form $a_nx^n + a_{n-1}x^{n-1} + \dots + a_1x + a_0 = 0$ (where the a_i are real numbers) that have no solutions. Somehow the most fundamental such example is the equation $x^2 + 1 = 0$, which has no solutions, because the square of any real number must be non-negative, and so $\sqrt{-1} \notin \mathbb{R}$.

It turns out that if we just throw in this one extra number $i := \sqrt{-1}$ to \mathbb{R} then we can solve *any* polynomial — a theorem so important that it’s known as the **fundamental theorem of algebra**. We call the result of doing this the **complex numbers**, and denote them by \mathbb{C} .

This gives us an algebraic way of understanding what a complex number is: it is a real number x plus an **imaginary** number iy (where $y \in \mathbb{R}$). That is, every complex number $x+iy$ simply corresponds to a pair of real numbers (x, y) . So now we can think geometrically! We imagine the complex numbers \mathbb{C} as the 2-dimensional Euclidean space \mathbb{R}^2 , where the x -axis corresponds to the real part of a complex number, and the y -axis to the imaginary part. This really is a geometric way of thinking, since now

To explain why we care so much about polynomials would be the subject of a whole nother book, but one important reason (of the *many!*), for both analysts and geometers alike, is the [Weierstrass Approximation Theorem](#).

addition (and subtraction) of complex numbers (which is defined by adding their real and imaginary parts separately) is given by vector addition, as shown in Figure 0.1.

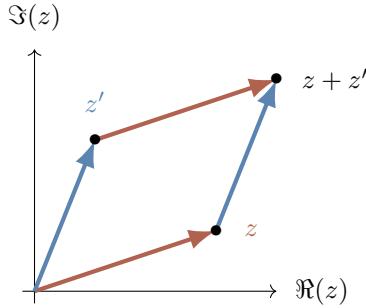


Figure 0.1: Addition of two complex numbers $z = x + iy$ and $z' = x' + iy'$, where we write Re (resp. Im) for the real (resp. imaginary) part of a complex number: $\text{Re}(x + iy) = x$ and $\text{Im}(x + iy) = y$. Commutativity of addition corresponds to what is sometimes called the **parallelogram law** for addition of vectors.

But what about multiplication and division? Following the rules of the game, we can figure out what the product of two complex numbers is by treating the imaginary number i as a “formal variable”, i.e. pretending it’s just a variable in some polynomial, and then remembering that $i = \sqrt{-1}$ at the very end:

$$\begin{aligned} (x + iy)(x' + iy') &= xx' + ixy' + iyx' + i^2yy' \\ &= xx' + ixy' + iyx' - yy' \\ &= xx' - yy' + i(xy' + yx'). \end{aligned}$$

Division works similarly — the most simple example of inverting a complex number $x + iy$ makes sense whenever x and y are both non-zero, since then we can use the trick of “multiplying by 1”:

$$\begin{aligned} \frac{1}{x + iy} &= \frac{1}{x + iy} \frac{x - iy}{x - iy} \\ &= \frac{x - iy}{x^2 + y^2} \\ &= \frac{x}{x^2 + y^2} - i \frac{y}{x^2 + y^2} \end{aligned}$$

This other complex number $x - iy$ that we used is somehow special because it is exactly the thing we needed to make the denominator real, so we give it a name: the **complex conjugate** of a complex number $z = x + iy$ is the complex number $z^* := x - iy$. Geometrically, this is just the reflection of the vector $(x, y) \in \mathbb{R}^2$ in the x -axis. The product $zz^* = x^2 + y^2$ is also important: you might recognise (from Pythagoras’ theorem) that $\sqrt{x^2 + y^2}$ is exactly the length of the vector (x, y) , and so we call the real number $|z| := \sqrt{zz^*}$ the **modulus** (or magnitude, norm, or absolute value). Note then that we can simply write $1/z = z^*/|z|^2$.

The more common notation in mathematics is \bar{z} instead of z^* , but physicists tend to like the latter.

Now things are looking somewhat nice, but the story isn’t complete. We have a good geometric intuition for what a complex number is (a vector in \mathbb{R}^2) and how to add them (vector addition), as well as what the complex conjugate and the modulus mean (reflection in the x -axis, and the length of the vector, respectively); but what about multiplication and division?

To understand these we need to switch from our **rectangular coordinates** $z = x + iy$ to **polar coordinates** — instead of describing a point z in \mathbb{R}^2 as “ x units left/right and y units up/down”, we describe it as “ r units from the origin, at an angle of θ **radians**”. We already know, given $(x, y) \in \mathbb{R}^2$, how to calculate its distance r from the origin, since this is exactly the length of the vector: $r = |(x, y)| = \sqrt{x^2 + y^2}$.

But what about the angle? Some trigonometry tells us that $\theta = \arctan(y/x)$, so we now know how to convert rectangular to polar coordinates:

$$x + iy = (x, y) \mapsto (r, \theta) := (\sqrt{x^2 + y^2}, \arctan(y/x)).$$

It would be nice to know how to go in the other direction though, but this can also be solved with some trigonometry:

$$(r, \theta) \mapsto (r \cos \theta, r \sin \theta).$$

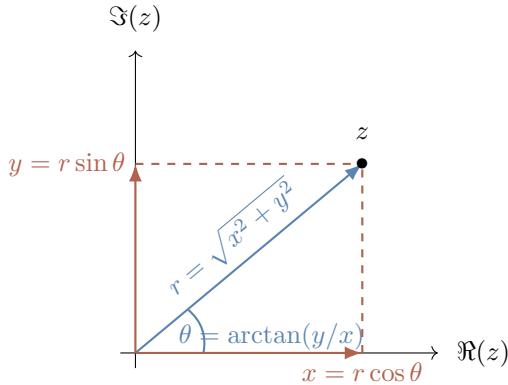


Figure 0.2: Expressing a complex number z in both planar and polar forms.

Great! ... but what's the point of polar coordinates? Well, it turns out that they give us a geometric way of understanding multiplication: you can show that (r, θ) multiplied by (r', θ') is exactly $(rr', \theta + \theta')$, which says that multiplication by a complex number (r, θ) is exactly a scaling by a factor of r and a rotation by θ . This means that we can also easily find the multiplicative inverse of (r, θ) , since it's just $(1/r, -\theta)$. Finally, complex conjugation just means switching the sign of the angle: $(r, \theta)^* = (r, -\theta)$.

There is one last ingredient that we should mention, which is the thing that really solidifies the relation between rectangular and polar coordinates. We know that rectangular coordinates (x, y) can be written as $x + iy$, so is there some more algebraic way of writing polar coordinates (r, θ) ? Then we can avoid any ambiguity that might arise from using pairs of numbers — if I tell you that I'm thinking of the complex number $z = (0.3, 2)$, do I mean the point $0.3 + 2i$, or the point that is distance r from the origin at an angle of 2 radians?

Given polar coordinates (r, θ) , we know that this is equal to $(r \cos \theta, r \sin \theta)$ in rectangular coordinates. For simplicity, let's first consider the case where $r = 1$. Then we can write $(1, \theta)$ as $\cos \theta + i \sin \theta$. Using the [Taylor series](#) of \sin and \cos , we can rewrite this as

$$\begin{aligned} \cos \theta + i \sin \theta &= \left(1 - \frac{\theta^2}{2!} + \frac{\theta^4}{4!} - \dots\right) + i \left(\theta - \frac{\theta^3}{3!} + \frac{\theta^5}{5!} - \dots\right) \\ &= 1 + i\theta - \frac{\theta^2}{2!} - i\frac{\theta^3}{3!} + \frac{\theta^4}{4!} + i\frac{\theta^5}{5!} - \dots \\ &= 1 + i\theta + \frac{i^2\theta^2}{2!} + \frac{i^3\theta^3}{3!} + \frac{i^4\theta^4}{4!} + \frac{i^5\theta^5}{5!} + \dots \\ &= \exp(i\theta) \end{aligned}$$

where at the very end we use the Taylor expansion of the [exponential function](#) $\exp(x) = e^x$.

We have just “proved” one of the most remarkable formulas in mathematics: Eu-

Exercise. Prove this!

If you don't know about Taylor series, then feel free to just skim this part, but make sure to read the punchline!

It is very important to point out that this “proof” is not rigorous or formal — you need to be very *very* careful when rearranging infinite sums! However, this proof *can be made rigorous* by using some real analysis.

ler's formula

$$e^{i\theta} = \cos \theta + i \sin \theta$$

(a special case of which gives the famous equation $e^{i\pi} + 1 = 0$, uniting five fundamental constants: 0, 1, i , e , and π). In summary then, we have two beautiful ways of expressing a complex number $z \in \mathbb{C}$, in either its rectangular/planar form or its polar/Euler form:

$$z = x + iy = re^{i\theta}.$$

Addition and subtraction are most neatly expressed in the planar form $x + iy$, and multiplication and division are most neatly expressed in the polar form $re^{i\theta}$; complex conjugation looks nice and tidy in both.

Addition of polar vectors.

We know how to perform addition, multiplication, inversion (which is a special case of division), and complex conjugation on complex numbers in planar form, but we've only described how to do the last *three* of these in polar form: we haven't said how to write $re^{i\theta} + r'e^{i\theta'}$ as $se^{i\varphi}$ for some s and φ . This is because it is very messy looking:

$$\begin{aligned}s &= \sqrt{r^2 + (r')^2 + 2rr' \cos(\theta' - \theta)} \\ \varphi &= \theta + \text{atan2}(r' \sin(\theta' - \theta), r + r' \cos(\theta' - \theta))\end{aligned}$$

and where `atan2` is the [2-argument arctangent function](#).

You do not need to know everything about this whole story of algebraically closed fields and so on, but it helps to know the basics, so here are some exercises that should help you to become more familiar.

- a. The set \mathbb{Q} of rational numbers and the set \mathbb{R} of real numbers are both fields, but the set \mathbb{Z} of integers is not. Why not?
- b. Look up the formal statement of the fundamental theorem of algebra.
- c. Evaluate each of the following quantities:

$$1 + e^{-i\pi}, \quad |1 + i|, \quad (1 + i)^{42}, \quad \sqrt{i}, \quad 2^i, \quad i^i.$$

- d. Here is a simple “proof” that $+1 = -1$:

$$1 = \sqrt{1} = \sqrt{(-1)(-1)} = \sqrt{-1}\sqrt{-1} = i^2 = -1.$$

What is wrong with it?

- e. Prove that, for any two complex numbers $w, z \in \mathbb{C}$, we always have the inequality

$$|z - w| \geq |z| - |w|.$$

- f. Using the fact that $e^{3i\theta} = (e^{i\theta})^3$, derive a formula for $\cos 3\theta$ in terms of $\cos \theta$ and $\sin \theta$.

Note that we have not really given you enough information in this section to be able to solve all these exercises, but that is intentional! Sometimes we like to ask questions and not answer them, with the hope that you will enjoy getting to do some research by yourself.

Hint: use polar form, draw a diagram, and appeal to the triangle inequality.

0.2 Euclidean vectors and vector spaces

We assume that you are familiar with Euclidean vectors — those arrow-like geometric objects which are used to represent physical quantities, such as trajectories, velocities,

or forces. You know that any two velocities can be added to yield a third, and the multiplication of a “velocity vector” by a real number is another “velocity vector”. So a **linear combination** of vectors is another vector: if v and w are vectors, and λ and μ are numbers (rational, real, or complex, for example), then $\lambda v + \mu w$ is another vector. Mathematicians have simply taken these properties and defined vectors as *anything* that we can add and multiply by numbers, as long as everything behaves in a nice enough way. This is basically what an Italian mathematician Giuseppe Peano (1858–1932) did in a chapter of his 1888 book with an impressive title: *Calcolo geometrico secondo l'Ausdehnungslehre di H. Grassmann preceduto dalle operazioni della logica deduttiva*. Following Peano, we define a **vector space** as a mathematical structure in which the notion of linear combination “makes sense”.

More formally, a **complex vector space** is a set V such that, given any two **vectors** a and b (that is, any two elements of V) and any two **complex** numbers α and β , we can form the linear combination $\alpha a + \beta b$, which is also a vector in V . There are certain “nice properties” that vector spaces things must satisfy. Addition of vectors must be commutative and associative, with an identity (the zero vector, which is often written as 0) and an inverse for each v (written as $-v$). Multiplication by complex numbers must obey the two distributive laws: $(\alpha + \beta)v = \alpha v + \beta v$ and $\alpha(v + w) = \alpha v + \alpha w$.

Modules over a ring.

A more succinct way of defining a vector space is as an abelian group endowed with a scalar action of a field. This showcases vector spaces as a particularly well behaved example of a more general object: **modules over a ring**.

A **subspace** of V is any subset of V which is closed under vector addition and multiplication by complex numbers. Here we start using the Dirac bra-ket notation and write vectors in a somewhat fancy way as $|label\rangle$, where the “label” is anything that serves to specify what the vector is. For example, $|\uparrow\rangle$ and $|\downarrow\rangle$ may refer to an electron with spin up or down along some prescribed direction, and $|0\rangle$ and $|1\rangle$ may describe a quantum bit holding either logical 0 or 1. As a maybe more familiar example, the set of binary strings of length n is a vector space over the field $\mathbb{Z}/2\mathbb{Z}$ of integers mod 2; in the case $n = 2$ we can write down all the vectors in this vector space in this notation: $|00\rangle, |01\rangle, |10\rangle, |11\rangle$, where e.g. $|10\rangle + |11\rangle = |01\rangle$ (addition is taken mod 2). These are often called **ket** vectors, or simply **kets**. (We will deal with “bras” in a moment).

A **basis** in V is a collection of vectors $|e_1\rangle, |e_2\rangle, \dots, |e_n\rangle$ such that every vector $|v\rangle$ in V can be written (in *exactly* one way) as a linear combination of the basis vectors: $|v\rangle = \sum_{i=1}^n v_i |e_i\rangle$, where the v_i are complex numbers. The number of elements in a basis is called the **dimension** of V . The most common, and prototypical, n -dimensional complex vector space (and the space which we will be using most of the time) is the space of ordered n -tuples of complex numbers, usually written as column vectors:

$$|a\rangle = \begin{bmatrix} a_1 \\ a_2 \\ \vdots \\ a_n \end{bmatrix}$$

with a basis given by the column vectors $|e_i\rangle$ that are 0 in every row except for a 1 in the i -th row:

$$|e_1\rangle = \begin{bmatrix} 1 \\ 0 \\ \vdots \\ 0 \end{bmatrix} \quad |e_2\rangle = \begin{bmatrix} 0 \\ 1 \\ \vdots \\ 0 \end{bmatrix} \quad \dots \quad |e_n\rangle = \begin{bmatrix} 0 \\ 0 \\ \vdots \\ 1 \end{bmatrix}$$

Showing that this definition is independent of the basis that we choose is a “fun” linear algebra exercise.

and where addition of vectors is done **component-wise**, so that

$$\left(\sum_{i=1}^n v_i |e_i\rangle \right) + \left(\sum_{i=1}^n w_i |e_i\rangle \right) = \sum_{i=1}^n (v_i + w_i) |e_i\rangle$$

or, in column vectors,

$$|v\rangle = \begin{bmatrix} v_1 \\ v_2 \\ \vdots \\ v_n \end{bmatrix} \quad |w\rangle = \begin{bmatrix} w_1 \\ w_2 \\ \vdots \\ w_n \end{bmatrix}$$

$$\alpha|a\rangle + \beta|b\rangle = \begin{bmatrix} \alpha v_1 + \beta w_1 \\ \alpha v_2 + \beta w_2 \\ \vdots \\ \alpha v_n + \beta w_n \end{bmatrix}$$

Throughout the course we will deal only with vector spaces of *finite* dimensions. This is sufficient for all our purposes and we will avoid many mathematical subtleties associated with infinite dimensional spaces, for which we would need the tools of **functional analysis**.

Formally, whenever we say *n-dimensional Euclidean space*, we mean the *real* vector space \mathbb{R}^n .

0.3 Bras and kets

An **inner product** on a vector space V (over the complex numbers) is a function that assigns to each pair of vectors $|u\rangle, |v\rangle \in V$ a complex number $\langle u|v\rangle$, and satisfies the following conditions:

- $\langle u|v\rangle = \langle v|u\rangle^*$
- $\langle v|v\rangle \geq 0$ for all $|v\rangle$
- $\langle v|v\rangle = 0$ if and only if $|v\rangle = 0$

where $*$ denotes complex conjugation (sometimes written as $z \mapsto \bar{z}$ instead).

The inner product must also be *linear* in the second argument but *antilinear* in the first argument:

$$\langle c_1 u_1 + c_2 u_2 | v \rangle = c_1^* \langle u_1 | v \rangle + c_2^* \langle u_2 | v \rangle$$

for any complex constants c_1 and c_2 .

To any physical system we associate a complex vector space with an inner product, known as a **Hilbert space** \mathcal{H} . The inner product between vectors $|u\rangle$ and $|v\rangle$ in \mathcal{H} is written as $\langle u|v\rangle$.

Finite-dimensional functional analysis.

If V is a vector space with an inner product $\langle -, - \rangle$, then this gives us a **norm** on V by defining $\|x\| = \sqrt{\langle x, x \rangle}$ and thus a **metric** by defining $d(x, y) = \|x - y\|$. We say that a sequence (x_n) in V is **Cauchy** if its elements “eventually always get closer”, i.e. if for all $\varepsilon > 0$ there exists some $N \in \mathbb{N}$ such that for all $m, n > N$ we have $\|x_n - x_m\| < \varepsilon$. We say that a normed space is **complete** if every Cauchy sequence converges in that space, i.e. if the limits of sequences that should exist actually do exist.

Now one useful fact is the following: on a *finite dimensional* vector space, all norms are equivalent. (Note that this does *not* mean that $\|x\|_1 = \|x\|_2$)

The question of *how* exactly we construct this associated space is a subtle one in the case of arbitrary physical systems, but we shall see that this is relatively straightforward when working with the types of systems that we consider in this book.

for any two norms $\| - \|_1$ and $\| - \|_2$, but simply that they “induce the same topology”—feel free to look up the precise definition elsewhere). This follows from another useful fact: in a *finite dimensional* vector space, the unit ball is compact. By a short topological argument, we can use these facts to show that what we claimed, namely that every *finite dimensional* inner product space is complete (with respect to the norm induced by the inner product, and thus with respect to *any* norm, since all norms are equivalent).

In the infinite dimensional case these facts are *not* true, and we have a special name for those inner product spaces which *are* complete: **Hilbert spaces**. So working in the finite dimensional case means that “we do not have to worry about analysis”, in that the completeness property comes for free the moment we have an inner product, and we can freely refer to inner product spaces as Hilbert spaces.

For example, for column vectors $|u\rangle$ and $|v\rangle$ in \mathbb{C}^n written as

$$|u\rangle = \begin{bmatrix} u_1 \\ u_2 \\ \vdots \\ u_n \end{bmatrix} \quad |v\rangle = \begin{bmatrix} v_1 \\ v_2 \\ \vdots \\ v_n \end{bmatrix}$$

their inner product is defined by

$$\langle u|v\rangle = u_1^*v_1 + u_2^*v_2 + \dots + u_n^*v_n.$$

Following Dirac, we may split the inner product into two ingredients:

$$\langle u|v\rangle \longrightarrow \langle u| |v\rangle.$$

Here $|v\rangle$ is a ket vector, and $\langle u|$ is called a **bra** vector, or a **bra**, and can be represented by a row vector:

$$\langle u| = [u_1^*, \ u_2^*, \ \dots, \ u_n^*].$$

The inner product can now be viewed as the result of the matrix multiplication:

$$\begin{aligned} \langle u|v\rangle &= [u_1^*, \ u_2^*, \ \dots, \ u_n^*] \cdot \begin{bmatrix} v_1 \\ v_2 \\ \vdots \\ v_n \end{bmatrix} \\ &= u_1^*v_1 + u_2^*v_2 + \dots + u_n^*v_n. \end{aligned}$$

Bras are vectors: you can add them, and multiply them by scalars (which, here, are complex numbers), but they are vectors in the space \mathcal{H}^* which is **dual** to \mathcal{H} . Elements of \mathcal{H}^* are **linear functionals**, that is, linear maps from \mathcal{H} to \mathbb{C} . A linear functional $\langle u|$ acting on a vector $|v\rangle$ in \mathcal{H} gives a complex number $\langle u|v\rangle$.

All Hilbert spaces of the same (finite) dimension are isomorphic, so the differences between quantum systems cannot be really understood without additional structure. This structure is provided by a specific algebra of operators acting on \mathcal{H} .

0.4 Daggers

Although \mathcal{H} and \mathcal{H}^* are not identical spaces — the former is inhabited by kets, and the latter by bras — they are closely related. There is a bijective map from one to the other given by $|v\rangle \leftrightarrow \langle v|$, and denoted by a **dagger**:

$$\begin{aligned}\langle v| &= (|v\rangle)^\dagger \\ |v\rangle &= (\langle v|)^\dagger.\end{aligned}$$

We usually omit the parentheses when it is obvious what the dagger operation applies to.

The dagger operation, also known as **Hermitian conjugation**, is *antilinear*:

$$\begin{aligned}(c_1|v_1\rangle + c_2|v_2\rangle)^\dagger &= c_1^*\langle v_1| + c_2^*\langle v_2| \\ (c_1\langle v_1| + c_2\langle v_2|)^\dagger &= c_1^*|v_1\rangle + c_2^*|v_2\rangle.\end{aligned}$$

Also, when applied twice, the dagger operation is the identity map.

You might already be familiar with Hermitian conjugation under another name: the **conjugate transpose** of an $(n \times m)$ matrix A is an $(m \times n)$ matrix A^\dagger , obtained by interchanging the rows and columns of A and taking complex conjugates of each entry in A , i.e. $A_{ij}^\dagger = A_{ji}^*$. In particular then,

$$|v\rangle = \begin{bmatrix} v_1 \\ v_2 \\ \vdots \\ v_n \end{bmatrix} \xleftarrow{\dagger} \langle v| = [v_1^*, \quad v_2^*, \quad \dots, \quad v_n^*].$$

We will come back to this \dagger operation on matrices in Section 0.6.

0.5 Geometry

The inner product brings geometry: the **length**, or **norm**, of $|v\rangle$ is given by $\|v\| = \sqrt{\langle v|v\rangle}$, and we say that $|u\rangle$ and $|v\rangle$ are **orthogonal** if $\langle u|v\rangle = 0$. Any maximal set of pairwise orthogonal vectors of unit length forms an orthonormal basis $\{|e_1\rangle, \dots, |e_n\rangle\}$, and so any vector can be expressed as a linear combination of the basis vectors:

$$|v\rangle = \sum_i v_i |e_i\rangle$$

where $v_i = \langle e_i|v\rangle$. Then the bras $\langle e_i|$ form the **dual basis**

$$\langle v| = \sum_i v_i^* \langle e_i|$$

where $v_i^* = \langle v|e_i\rangle$.

To make the notation a bit less cumbersome, we will sometimes label the basis kets as $|i\rangle$ rather than $|e_i\rangle$, and write

$$\begin{aligned}|v\rangle &= \sum_i |i\rangle \langle i|v\rangle \\ \langle v| &= \sum_j \langle v|i\rangle \langle i|\end{aligned}$$

but do not confuse $|0\rangle$ with the zero vector! We never write the zero vector as $|0\rangle$, but only ever as 0, without any bra or ket decorations (so e.g. $|v\rangle + 0 = |v\rangle$).

Now that we have some notion of geometry, we can explain a bit more about this idea of associating a Hilbert space to a quantum system — we will use some terminology that we have not yet introduced, but all will be explained in due time.

“Is this a \dagger which I see before me...”

In mathematics texts this operation is often denoted by $*$ rather than \dagger , but we reserve the former for complex conjugation *without* matrix transposition. Note, however, that scalars can be thought of as (1×1) matrices, and in this special case we have that $\dagger = *$.

That is, consider sets of vectors $|e_i\rangle$ such that $\langle e_i|e_j\rangle = \delta_{ij}$ (where the **Kronecker delta** δ_{ij} is 0 if $i \neq j$, and 1 if $i = j$.), and then pick any of the largest such sets (which must exist, since we assume our vector spaces to be finite dimensional).

To any *isolated* quantum system, which can be prepared in n **perfectly distinguishable** states, we can associate a Hilbert space \mathcal{H} of dimension n such that each vector $|v\rangle \in \mathcal{H}$ of unit length $\langle v|v\rangle = 1$ represents a quantum state of the system. The overall phase of the vector has no physical significance: $|v\rangle$ and $e^{i\varphi}|v\rangle$ (for any real φ) both describe the same state.

We note here one more fact that also won't yet make sense, but which won't hurt to have hidden away in the back of your mind.

The inner product $\langle u|v\rangle$ is the **probability amplitude** that a quantum system prepared in state $|v\rangle$ will be found in state $|u\rangle$ upon measurement. This means that states corresponding to orthogonal vectors (i.e. $\langle u|v\rangle = 0$) are perfectly distinguishable: if we prepare the system in state $|v\rangle$, then it will never be found in state $|u\rangle$, and vice versa.

0.6 Operators

A **linear map** between two vector spaces \mathcal{H} and \mathcal{K} is a function $A: \mathcal{H} \rightarrow \mathcal{K}$ that respects linear combinations:

$$A(c_1|v_1\rangle + c_2|v_2\rangle) = c_1A|v_1\rangle + c_2A|v_2\rangle$$

for any vectors $|v_1\rangle, |v_2\rangle$ and any complex numbers c_1, c_2 . We will focus mostly on **endomorphisms**, that is, maps from \mathcal{H} to \mathcal{H} , and we will call them **operators**. The symbol $\mathbf{1}$ is reserved for the identity operator that maps every element of \mathcal{H} to itself (i.e. $\mathbf{1}|v\rangle = |v\rangle$ for all $|v\rangle \in \mathcal{H}$). The product BA of two operators A and B is the operator obtained by first applying A to some ket $|v\rangle$ and then B to the ket which results from applying A :

$$(BA)|v\rangle = B(A|v\rangle).$$

The order does matter: in general, $BA \neq AB$. In the exceptional case in which $AB = BA$, one says that these two operators **commute**. The inverse of A , written as A^{-1} , is the operator that satisfies $AA^{-1} = \mathbf{1} = A^{-1}A$. For finite-dimensional spaces, one only needs to check *one* of these two conditions, since any one of the two implies the other, whereas, on an infinite-dimensional space, *both* must be checked. Finally, given a particular basis, an operator A is uniquely determined by the entries of its matrix: $A_{ij} = \langle i|A|j\rangle$.

The **adjoint**, or **Hermitian conjugate**, of a linear map A , denoted by A^\dagger , is defined by the relation

$$\begin{aligned} \langle i|A^\dagger|j\rangle &= \langle j|A|i\rangle^* \\ \text{for all } |i\rangle, |j\rangle \in \mathcal{H} \end{aligned}$$

and \dagger turns $(n \times m)$ matrices into $(m \times n)$ matrices.

An operator A is said to be

- **normal** if $AA^\dagger = A^\dagger A$
- **unitary** if $A^\dagger = A^{-1}$
- **Hermitian** (or **self-adjoint**) if $A^\dagger = A$.

In particular then, being unitary implies being normal, since if $A^\dagger = A^{-1}$ then $AA^\dagger = A^\dagger A$, since both of these are equal to $\mathbf{1}$. Note also that unitary and Hermitian operators must indeed be *operators*, i.e. they are represented by a *square* matrix.

Any *physically admissible* evolution of an isolated quantum system is represented by a unitary operator. Note that unitary operators preserve the inner product: given a unitary operator U and two kets $|a\rangle$ and $|b\rangle$, and defining $|a'\rangle = U|a\rangle$ and $|b'\rangle = U|b\rangle$, we have that

$$\begin{aligned}\langle a' | &= \langle a | U^\dagger \\ \langle b' | &= \langle b | U^\dagger \\ \langle a' | b' \rangle &= \langle a | U^\dagger U | b \rangle = \langle a | \mathbf{1} | b \rangle = \langle a | b \rangle.\end{aligned}$$

Preserving the inner product implies preserving the norm induced by this product, i.e. unit state vectors are mapped to unit state vectors, i.e. *unitary operations are the isometries of the Euclidean norm*.

This is an *axiom*, justified by experimental evidence, and also by some sort of mathematical intuition. So, in this book, we take this as a *fact* that we do not question. It is, however, very interesting to question it: *why should we assume this to be true?*

Dagger compact categories.

This whole package of stuff and properties and structure (i.e. finite dimensional Hilbert spaces with linear maps and the dagger) bundles up into an abstract framework called a **dagger compact category**. We will not delve into the vast world of category theory in this book, and to reach an understanding of all the ingredients that go into the one single definition of dagger compact categories would take more than a single chapter. But it's a good idea to be aware that there are researchers in quantum information science who work *entirely* from this approach, known as **categorical quantum mechanics**.

One particular method within this approach is the use of **string diagrams**, which allow for the use of so-called diagrammatic reasoning, with the **ZX-calculus** being a particularly successful example. For an introduction to string diagrams of this flavour, it's maybe a good idea to start with understanding how they can express the linear algebra that you already know. For example, Paweł Sobociński's "**Graphical Linear Algebra**" aims to teach linear algebra entirely through the introduction of string diagrams.

0.7 Eigenvalues and eigenvectors

Given an operator A , an **eigenvector** is a non-zero vector $|v\rangle$ such that

$$A|v\rangle = \lambda|v\rangle$$

for some $\lambda \in \mathbb{C}$ (which is called the corresponding **eigenvalue**). We call the pair $(\lambda, |v\rangle)$ an **eigenpair**, and we call the set of eigenvalues the **spectrum** of A , denoted by $\sigma(A)$. It is a surprising (but incredibly useful) fact that every operator has at least one eigenpair. Geometrically, an eigenvector of an operator A is a vector upon which A simply acts by “stretching”. Note that eigenvectors can be scaled by an arbitrary non-zero length: if $A|v\rangle = \lambda|v\rangle$ then

$$\begin{aligned}A(\mu|v\rangle) &= \mu(A|v\rangle) \\ &= \mu\lambda|v\rangle \\ &= \lambda(\mu|v\rangle)\end{aligned}$$

for any $\mu \neq 0$. Because of this, we usually assume all eigenvectors to be of length 1.

Rewriting the defining property of an eigenpair $(\lambda, |v\rangle)$, we see that

$$(A - \lambda\mathbf{1})|v\rangle = 0$$

which tells us that the operator $A - \lambda\mathbf{1}$ has a non-zero kernel, and is thus non-invertible. This gives a useful characterisation of the spectrum in terms of a determinant:

$$\sigma(A) = \{\lambda \in \mathbb{C} \mid \det(A - \lambda\mathbf{1}) = 0\}.$$

You can prove this for an $(n \times n)$ matrix A by considering the set $\{|v\rangle, A|v\rangle, A^2|v\rangle, \dots, A^n|v\rangle\}$ of vectors in \mathbb{C}^n . Since this has $n + 1$ elements, it must be linearly *dependent*, and so (after some lengthy algebra) we can construct an eigenpair.

The spectrum $\sigma(A)$ allows us to recover information about the operator A . For example, the trace of A is equal to the sum of all its eigenvalues, and the determinant of A is equal to the product of all its eigenvalues. We can show this easily for normal operators using the fact that their eigenvectors are orthogonal: they satisfy $\langle v|w \rangle = 0$ for $v \neq w$. Because eigenvectors can always be scaled, this means that we can assume the eigenvectors of a normal operator to be *orthonormal*. If we write the eigenpairs as $(\lambda_i, |v_i\rangle)$, we can define

$$U = \sum_i |i\rangle\langle v_i|$$

for an orthonormal basis $\{|1\rangle, \dots, |n\rangle\}$, which is an orthogonal matrix (since the eigenvectors are also assumed to be orthonormal). Then we see that

$$\begin{aligned} UAU^\dagger &= \sum_{i,j} |i\rangle\langle v_i|A|v_j\rangle\langle j| \\ &= \sum_{i,j} |i\rangle\langle v_i|\lambda_j|v_j\rangle\langle j| \\ &= \sum_{i,j} \lambda_j|i\rangle(\langle v_i|v_j\rangle)\langle j| \\ &= \sum_i \lambda_i|i\rangle\langle i| \end{aligned}$$

(where we again use this hypothesis that the eigenvectors of A are all orthonormal) which is the diagonal matrix D consisting of the eigenvalues λ_i of A along its diagonal. Then, since $UAU^\dagger = D$, we can equally write $A = U^\dagger DU$, which gives us

$$A = \sum_i \lambda_i |v_i\rangle\langle v_i|.$$

We call each $\lambda_i |v_i\rangle\langle v_i|$ the **eigenspace projector**, since a **projector** is defined to be any operator P that satisfies $P = P^\dagger$ and $P^2 = P$. Note that projectors can only have eigenvalues equal to 0 or 1, since if $|v\rangle$ is an eigenvector of P then, using the fact that $P^2 = P$,

$$\begin{aligned} 0 &= (P^2 - P)|v\rangle \\ &= (\lambda^2 - \lambda)|v\rangle \\ \implies \lambda(\lambda - 1) &= 0 \end{aligned}$$

and so λ must be equal to either 0 or 1.

Finally we can now return to the relationship between eigenvalues and the trace and determinant, using this fact that any normal operator A gives a unitary operator U such that $A = U^\dagger DU$ for the diagonal matrix D of eigenvalues of A . Indeed,

$$\begin{aligned} \text{tr}(A) &= \text{tr}(U^\dagger DU) \\ &= \text{tr}(DUU^\dagger) \\ &= \text{tr}(D) \\ &= \sum_i \lambda_i \end{aligned}$$

Exercise. Prove this! Hint: start by showing that, if $(\lambda, |v\rangle)$ is an eigenpair of A , then $(\lambda^*, |v\rangle)$ is an eigenpair of A^\dagger .

which proves that the trace is equal to the sum of the eigenvalues, and

$$\begin{aligned}\det(A) &= \det(U^\dagger D U) \\ &= \det(U^\dagger) \det(D) \det(U) \\ &= \det(D) \det(U^\dagger) \det(U) \\ &= \det(D) \det(U^\dagger U) \\ &= \det(D) \\ &= \prod_i \lambda_i\end{aligned}$$

which proves that the determinant is equal to the product of the eigenvalues.

The eigenspace projectors give us the **spectral decomposition** of A , which is where we write

$$A = \sum_i \lambda_i |v_i\rangle\langle v_i|.$$

Extending functions to matrices.

The spectral decomposition of a normal operator gives an effective way of calculating the action of a function on a matrix. If $f: \mathbb{C} \rightarrow \mathbb{C}$ is a function, then we can define

$$f(A) = \sum_i f(\lambda_i) |v_i\rangle\langle v_i|.$$

For example, if $f(x) = x^2$, then, by this definition, $f(A) = \sum_i \lambda_i^2 |v_i\rangle\langle v_i|$. But this is consistent with the definition of A^2 that you expect:

$$\begin{aligned}A^2 &= \left(\sum_i \lambda_i |v_i\rangle\langle v_i| \right) \left(\sum_i \lambda_i |v_i\rangle\langle v_i| \right) \\ &= \sum_{i,j} \lambda_i \lambda_j |v_i\rangle\langle v_i| |v_j\rangle\langle v_j| \\ &= \sum_i \lambda_i^2 |v_i\rangle\langle v_i|\end{aligned}$$

using the fact that the eigenvectors $|v_i\rangle$ are orthonormal and that projectors $P = |v_i\rangle\langle v_i|$ satisfy $P^2 = P$.

0.8 Outer products

Apart from the inner product $\langle u|v\rangle$, which is a complex number, we can also form the **outer product** $|u\rangle\langle v|$, which is a linear map (operator) on \mathcal{H} (or on \mathcal{H}^* , depending how you look at it). This is what physicists like (and what mathematicians dislike!) about Dirac notation: a certain degree of healthy ambiguity.

- The result of $|u\rangle\langle v|$ acting on a ket $|x\rangle$ is $|u\rangle\langle v|x\rangle$, i.e. the vector $|u\rangle$ multiplied by the complex number $\langle v|x\rangle$.
- Similarly, the result of $|u\rangle\langle v|$ acting on a bra $\langle y|$ is $\langle y|u\rangle\langle v|$, i.e. the linear functional $\langle v|$ multiplied by the complex number $\langle y|u\rangle$.

The product of two maps, $A = |a\rangle\langle b|$ followed by $B = |c\rangle\langle d|$, is a linear map BA , which can be written in Dirac notation as

$$BA = |c\rangle\langle d|a\rangle\langle b| = \langle d|a\rangle|c\rangle\langle b|$$

i.e. the inner product (complex number) $\langle d|a \rangle$ times the outer product (linear map) $|c\rangle\langle b|$.

Any operator on \mathcal{H} can be expressed as a sum of outer products. Given an orthonormal basis $\{|e_i\rangle\}_{i=1,\dots,n}$, any operator which maps the basis vectors $|e_i\rangle$ to vectors $|f_i\rangle$ can be written as $\sum_{i=1}^n |f_i\rangle\langle e_i|$. If the vectors $\{|f_i\rangle\}$ also form an orthonormal basis then the operator simply “rotates” one orthonormal basis into another. These are unitary operators which preserve the inner product. In particular, if each $|e_i\rangle$ is mapped to $|e_i\rangle$, then we obtain the identity operator:

$$\sum_i |e_i\rangle\langle e_i| = \mathbf{1}.$$

This relation holds for *any* orthonormal basis, and it is one of the most ubiquitous and useful formulas in quantum theory, known as **completeness**. For example, for any vector $|v\rangle$ and for any orthonormal basis $\{|e_i\rangle\}$, we have

$$\begin{aligned} |v\rangle &= \mathbf{1}|v\rangle \\ &= \sum_i |e_i\rangle\langle e_i| |v\rangle \\ &= \sum_i |e_i\rangle \langle e_i| v \\ &= \sum_i v_i |e_i\rangle \end{aligned}$$

where $v_i = \langle e_i|v\rangle$ are the components of $|v\rangle$.

Finally, note that calculating the adjoint of an outer product boils down to just swapping the order:

$$(|a\rangle\langle b|)^\dagger = |b\rangle\langle a|.$$

0.9 The trace

The **trace** is an operation which turns outer products into inner products,

$$\text{tr}: |b\rangle\langle a| \longmapsto \langle a|b\rangle.$$

We have just seen that any linear operator can be written as a sum of outer products, and so we can extend the definition of trace (by linearity) to any operator. Equivalently, for any square matrix A , the trace of A can be defined to be the sum of its diagonal elements:

$$\text{tr } A = \sum_k \langle e_k|A|e_k\rangle = \sum_k A_{kk}.$$

In fact, the trace of A is equal to the sum of the eigenvalues of A , even in the case where A is not diagonalisable.

You can show, using this definition or otherwise, that the trace is cyclic ($\text{tr}(AB) = \text{tr}(BA)$) and linear ($\text{tr}(\alpha A + \beta B) = \alpha \text{tr}(A) + \beta \text{tr}(B)$, where A and B are square matrices and α and β complex numbers).

To recover the first definition from the second, we argue as follows:

$$\begin{aligned} \text{tr } |b\rangle\langle a| &= \sum_k \langle e_k|b\rangle\langle a|e_k\rangle \\ &= \sum_k \langle a|e_k\rangle\langle e_k|b\rangle \\ &= \langle a|\mathbf{1}|b\rangle \\ &= \langle a|b\rangle. \end{aligned}$$

Not to be confused with “completeness” in the sense of Hilbert spaces.

Note that “cyclic” does not mean the same thing as “permutation invariant”! It is *not* true in general that $\text{tr}(ABC) = \text{tr}(CBA)$, but only that $\text{tr}(ABC) = \text{tr}(BCA) = \text{tr}(CAB)$, i.e. we can only *cyclically* permute the operators.

Here, the second term can be viewed both as the sum of the diagonal elements of $|b\rangle\langle a|$ in the $|e_k\rangle$ basis, and as the sum of the products of two complex numbers $\langle e_k|b\rangle$ and $\langle a|e_k\rangle$. We have used the decomposition of the identity, $\sum_k |e_k\rangle\langle e_k| = 1$. Given that we can decompose the identity by choosing any orthonormal basis, it is clear that the trace does *not* depend on the choice of the basis.

0.10 Some useful identities

Here is a summary of some particularly useful equalities concerning bras, kets, inner products, outer products, traces, and operators, that we will be using time and time again. In all of these, $|a\rangle, |b\rangle \in \mathcal{H}$ are kets, A, B, C are operators on \mathcal{H} , and $\alpha, \beta \in \mathbb{C}$ are scalars.

Dagger for bras and kets:

- $|a\rangle^\dagger = \langle a|$
- $\langle a|^\dagger = |a\rangle$
- $(|a\rangle\langle b|)^\dagger = |b\rangle\langle a|$
- $(\alpha|a\rangle + \beta|b\rangle)^\dagger = \alpha^*\langle a| + \beta^*\langle b|$

Dagger for operators:

- $(AB)^\dagger = B^\dagger A^\dagger$
- $(A^\dagger)^\dagger = A$
- $(\alpha A + \beta B)^\dagger = \alpha^* A^\dagger + \beta^* B^\dagger$

Trace:

- $\text{tr}(\alpha A + \beta B) = \alpha \text{tr}(A) + \beta \text{tr}(B)$
- $\text{tr}(ABC) = \text{tr}(CAB) = \text{tr}(BCA)$
- $\text{tr}|a\rangle\langle b| = \langle b|a\rangle$
- $\text{tr}(A|a\rangle\langle b|) = \langle b|A|a\rangle = \text{tr}(|a\rangle\langle b|A)$

0.11 Probabilities

In a sense, the basics of quantum theory boil down to the combination of two bits of mathematics: linear algebra over the complex numbers, and probability theory. We have just gone over all the linear algebra that we will need, so now let's tackle the other topic (though we will immediately revisit it in Chapter 1).

Probability theory is a vast and beautiful subject which has undergone many transformations over the centuries. What started as something understood in terms of gambling odds later evolved into the theory of measure spaces, and is now even able to be expressed in terms of diagrammatic category theory. But for our purposes, we only need the very elementary parts of the subject, so we will stick with the first interpretation: probability tells us the odds of something happening.

The setup is always the same: we have some process (rolling some dice, flipping a coin, drawing a card, etc.) that has some possible **outcomes** (getting a 5, heads, or an ace of hearts, etc.) but is realised in some way which means that we cannot be certain which outcome we will see whenever we run the process (or “perform the experiment”).

The first thing to define in any such scenario is the **sample space**, usually denoted by Ω , which is the set of all possible outcomes. Next we have the **event space** \mathcal{F} , which is the set of all **events**, where an event is a set of outcomes (this might sound

What we actually mean by “the odds of something happening” and how we should really interpret probabilities “in the real world” is a profound philosophical problem that we shall completely pass over.

confusing at first, but we'll give some examples shortly). Whenever we run the process, we will get some outcome, and we say that any event that contains that outcome **occurred**. Finally, we will have a **probability function**, which assigns to any event a **probability**, which is a number between 0 and 1. This probability function has to satisfy some axioms, but we'll come to these later; for now, let us give some examples. Here is a table of the sample spaces for some processes.

Process	Sample space Ω
Rolling a six-sided die	$\{1, 2, 3, 4, 5, 6\}$
Flipping a coin	$\{H, T\}$
Flipping two <i>distinct</i> coins	$\{HH, TH, HT, TT\}$
Flipping two <i>identical</i> coins	$\{HH, TH, TT\}$

And here's a table of some (but not all, except for in the case of flipping a single coin) of the events corresponding to these sample spaces.

Process	Example events $A \in \mathcal{F}$	Interpretation
Rolling a six-sided die	$\{1\}$	rolling a 1
	$\{1, 3, 5\}$	rolling an odd number
	$\{1, 2, 3\}$	rolling a number less than 3
	$\{2, 3\}$	rolling a prime number less than 4
Flipping a coin	$\{H\}$	getting heads
	$\{T\}$	getting tails
	$\{H, T\}$	any outcome at all
Flipping two <i>distinct</i> coins	$\{HH\}$	getting two heads
	$\{HH, TH, HT\}$	getting at least one heads
	$\{HH, TT\}$	getting two the same
Flipping two <i>identical</i> coins	$\{HH\}$	getting two heads
	$\{HH, TH\}$	getting at least one heads

In the table above we can see that, for example, if we rolled a 1 on a six-sided die then many events occurred: we rolled a 1, but we also rolled an odd number, and a number less than 3; but we did not roll a prime number.

Something else that arises in these examples the notion of **distinguishable outcomes**, when we look at how the sample space of flipping two coins depends on whether or not they are identical. That is, if we have a gold coin and a silver coin then it makes sense to say that HT is different from TH , because the first means that the gold coin was the one that landed on heads, and the second means that it was instead the silver coin. But if we have two identical coins, then how can we distinguish between HT and TH ? We would have to be able to point to one of them, say, the one on the left, and say “that's the coin that's on the left”, but if we can do this then by definition we can distinguish between them! In general, the sample space consists only of distinguishable outcomes, but what counts as distinguishable really depends on the specifics of the experiment that we have set up.

Another possibility would be to distinguish the coins *in time* instead of space, i.e. to flip one coin first and then the other afterwards. A coin cannot remember what happened the last time it was flipped, so is there really a difference between flipping a single coin twice or two coins once? In the eyes of probability theory, the answer is “no”.

Measure theory.

This approach towards probability, where we think of a “scenario” as such a triple (Ω, \mathcal{F}, P) , places us firmly within the setting of **measure theory**. What we have described is a **probability measure**, and there are many more general types of measure that exist, but they all describe the same sort of idea: we have a set Ω that we think of as our “space”, some particularly well-behaved collection \mathcal{F} of subsets of Ω , and a function $\mu: \mathcal{F} \rightarrow \mathbb{R} \sqcup \{\pm\infty\}$ known as the **measure**. We think of the measure μ as telling us how big any element of \mathcal{F} is, be it interpreted as geometric size or, in the example of probability measures, the likelihood. This formalism becomes very useful when we want to do any analysis (which happens as soon as we want to deal with infinite-dimensional spaces, or even just “introduce some geometry”), such as in constructing the **Lebesgue integral**. Thinking of probability spaces through measure theory allows us to make use of the many powerful tools of real analysis.

Now we can define probability rather succinctly.

In a **fair process**, where all outcomes are equally likely, the **probability** $P(A)$ of an event A is the number of desired outcomes divided by the number of total outcomes, i.e.

$$P(A) = \frac{\text{\# of elements of } A}{\text{\# of elements of } \Omega}.$$

In particular, the probability will always be a number between 0 and 1.

Running through some of the above examples of events, we see that this definition of probability agrees with what we might already expect.

Event	Probability
Getting heads on a single coin flip	1/2
Rolling a 6 with a single die	1/6
Rolling an odd number with a single die	3/6 = 1/2

Pascal’s triangle.

Flipping a fair coin (or actually, even an unfair one) is a common scenario in discussing probability, because it has just two outcomes — the smallest amount you can have without things becoming purely deterministic. There are lots of numbers that you will see turn up time and time again in calculations of probability for binary outcome events, and most usually they are **binomial coefficients**. These are numbers that can be read directly from the rows of **Pascal’s triangle** (which, as is often the case in mathematics, is more deserving of being named after a different person: [Al-Karaji](#), or maybe [Omar Khayyam](#)), and they satisfy many interesting combinatorial patterns.

Now let’s look at what happens when we’re interested in more than one event occurring. We might study the possibility of *either* event A or event B happening, where the “or” here can be **exclusive** (we want exactly one of them to happen, but *not* both) or **inclusive** (we want *at least* one of them to happen), or we could study the possibility of *both* event A and event B happening. It turns out that these two

notions, which seem somehow opposite, are delicately related.

First of all, let's consider *both* events A and B occurring. What does this mean? Well, by our definition of “occurring”, it means that the outcome of the process is an element of A and also of B , which is equivalent to saying that it is an element of their intersection $A \cap B$. In other words,

$$P(A \text{ and } B) = P(A \cap B).$$

This lets us define two very important terms.

We say that A and B are **mutually exclusive** if $P(A \cap B) = 0$, and **independent** if $P(A \cap B) = P(A)P(B)$.

In words, A and B are mutually exclusive if one of them occurring precludes the other from occurring, i.e. if *at most* one of them can occur, and they are independent if one of them occurring has no effect on the other occurring.

Usually, when we talk of mutually exclusive events we are referring to a single run of an experiment, and for independent events we are referring to multiple runs. For example, “rolling an even number” and “rolling an odd number” are mutually exclusive events when rolling a single die once, but independent events when rolling a single die twice. Basically, we should be careful when talking about events and make sure to be precise as to what our sample space is, and how the event is actually realised as a subset of this.

We can think of mutually exclusive and independent as extreme ends of a scale: on one side we have events that affect each other so strongly that if one occurs then we know with absolute certainty that the other one did not; on the other we have events that have absolutely no effect on each other whatsoever. One might wonder about what the opposite of mutually exclusive might be, and there are two ideas that seem like they might be interesting: events A and B such that $P(A \cap B) = 1$, and events A and B such that, if one occurs, then the other also always occurs. The first of these two putative definitions is not so interesting, because if $P(A \cap B) = 1$ then both A and B always occur, and so, by the definition of $P(A) = P(B) = 1$, this means that $A = B = \Omega$ is just the event that “anything happens”. The second is a bit less trivial, but also not so deep: saying that A occurs if and only if B occurs is equivalent to saying that $A = B$ as events.

Now let's think about *either* event A or event B occurring, in the inclusive-or sense of the word (we will return to the exclusive one afterwards). This is equivalent to the event given by the union $A \cup B$ occurring. In other words,

$$P(A \text{ or } B) = P(A \cup B).$$

The relationship between $P(A \cup B)$ and $P(A \cap B)$ is given by the following principle.

Inclusion-exclusion principle. $P(A \cup B) = P(A) + P(B) - P(A \cap B)$.

We will not prove this, but it's a fun exercise to think about why this must be true. In fact, the general inclusion-exclusion principle describes what happens for an arbitrary finite number of events. Using this, we can see why mutually independent events are particularly nice: if A and B are mutually exclusive, then $P(A \cup B) = P(A) + P(B)$, i.e. the probability of $(A \text{ or } B)$ is the sum of the probability of A and the probability of B .

In the same way that mutually exclusive events are special in the eyes of the inclusion-exclusion principle, independent events are special in the eyes of conditional probability. Oftentimes we consider events that are not independent, such as

Exercise. Are the events “rolling an even number” and “rolling an odd number” still independent when we think of rolling two die simultaneously?

Drawing a Venn diagram might help.

drawing cards from a deck without replacing them afterwards: if I take the ace of hearts, then the probability of me drawing a heart the next time has gone down from $13/52$ to $12/51$, since there is now one fewer heart in the deck. Even worse, the probability of me drawing the ace of hearts again is now 0. Given two events A and B , we define the **conditional probability** $P(B|A)$, of B given A , to be the probability that B occurs *assuming that A has just previously occurred*. Thinking back to the definition of probability, we can calculate this by taking the number of outcomes in $A \cap B$ (our desired outcomes, the outcomes in B that also are outcomes in A) and dividing it by the number of outcomes in A (all possible outcomes, since we are assuming that A has happened), so that

$$P(B|A) = \frac{P(A \cap B)}{P(A)}.$$

Conditional probabilities are the source of many misunderstandings. For example, it's intuitively obvious that the probability of flipping a coin 100 times and getting heads every single time is very small. So say we've flipped a coin 99 times and managed to get a heads every single time, are we now more likely to flip a tail, because the chance of getting 100 heads in a row must be small? Well we don't even need mathematics to answer this: the coin has no way of remembering what has happened on the previous flips! In other words, also the probability $P(H^{100}) = P(H^{99})$ and H is very small, the (conditional) probability $P(H|H^{99})$ is still exactly $1/2$. Looking at the definition of conditional probability, this makes sense: $P(H^{99})$ is itself very small, almost as small as $P(H^{100})$, so it's not surprising that when we divide one by the other we get a number that's not so far away from 1.

Now we can see how independent events are special: if $P(A \cap B) = P(A)P(B)$ then

$$\begin{aligned} P(B|A) &= \frac{P(A \cap B)}{P(A)} \\ &= \frac{P(A)P(B)}{P(A)} \\ &= P(B) \end{aligned}$$

and so the probability of B given A is exactly the same as the probability of B without knowing anything about the outcome of A (and similarly for $P(A|B) = P(A)$).

Finally, we mention what might be called the fundamental theorem of conditional probability.

Bayes' theorem. Let A and B be events with $P(B) \neq 0$. Then

$$P(A|B) = \frac{P(B|A)P(A)}{P(B)}$$

You should now be able to answer the following questions:

1. When you roll a normal six-sided die, what is the set of distinguishable outcomes?
2. What is the probability of getting a 5?
3. What is the probability of getting a number (strictly) less than 3?

Now imagine that you have two six-sided dice.

4. If you roll both dice at the same time, what is the probability of them both landing on a 6?
5. What is the probability of getting two numbers that add up to 6?

Finally, we give our two six-sided dice to a friend for them to roll in secret.

6. If they tell us that they rolled two numbers that added up to 6, what is the probability that they rolled a 1?

Diagrammatic probability theory.

We mentioned in Section 0.6 that a lot of the structure inherent in our formalism of quantum theory can be encapsulated by the notion of a dagger compact category, and can thus be investigated with a diagrammatic approach. It turns out that parts of probability theory — specifically [Markov processes](#), which describe scenarios where different events can happen with varying probabilities, but where nothing depends on the history of the scenario, only on the here-and-now — are also amenable to such an approach. This leads to the definition of a [Markov category](#).

Part I

Foundations

1 Quantum interference

*About complex numbers, called **probability amplitudes**, that, unlike probabilities, can cancel each other out, leading to **quantum interference**, and consequently qualitatively new ways of processing information.*

The classical theory of computation does not usually refer to physics. Pioneers such as Alan Turing, Alonzo Church, Emil Post, and Kurt Gödel managed to capture the correct classical theory by intuition alone and, as a result, it is often falsely assumed that its foundations are self-evident and purely abstract. They are not!

Possibly the most important motto of this book is the following: “*Computation is a physical process. Computation is a physical process. Computation is . . .*”

The concepts of information and computation can be properly formulated only in the context of a physical theory — information is stored, transmitted and processed always by *physical* means. Computers are physical objects and computation is a physical process. Indeed, any computation, classical or quantum, can be viewed in terms of physical experiments, which produce **outputs** that depend on initial preparations called **inputs**. Once we abandon the classical view of computation as a purely logical notion independent of the laws of physics it becomes clear that whenever we improve our knowledge about physical reality, we may also gain new means of computation. Thus, from this perspective, it is not very surprising that the discovery of quantum mechanics in particular has changed our understanding of the nature of computation. In order to explain what makes quantum computers so different from their classical counterparts, we begin with the rudiments of quantum theory.

Some of what we say in this chapter will be repeated in later chapters, but usually in much more detail. Feel free to think of this chapter as a sort of “aeroplane tour” of the rudiments, knowing that we will soon land on the ground to go out exploring by foot.

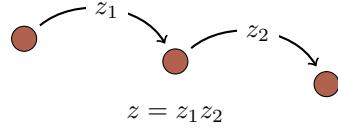
1.1 Two basic rules

Quantum theory, at least at some instrumental level, can be viewed as a modification of probability theory: we replace *positive real* numbers (i.e. probabilities) with *complex* numbers z , called **probability amplitudes** (or simply “amplitudes”), such that the squares of their absolute values $|z|^2$ are interpreted as probabilities.

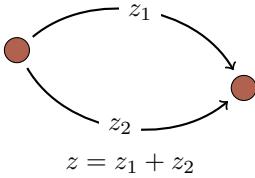
The correspondence between probability amplitudes z and probabilities $|z|^2$ is known as **Born’s rule**, named for physicist and mathematician [Max Born](#) (1882–1970).

The rules for combining amplitudes are very reminiscent of the rules for combining probabilities:

1. Whenever something can happen in a *sequence of independent* steps, we *multiply* the amplitudes of each step.



2. Whenever something can happen in *several alternative ways*, we *add* the amplitudes for each separate way.



That's it! These two rules are basically all you need to manipulate amplitudes in any physical process, no matter how complicated. They are universal and apply to any physical system, from elementary particles through atoms and molecules to white dwarfs stars. They also apply to information, since, as we have already emphasised, information is physical. The two rules look deceptively simple but, as you will see in a moment, their consequences are anything but trivial.

1.2 The failure of probability theory

Modern mathematical probability theory is based on three axioms, proposed by [Andrey Nikolaevich Kolmogorov](#) (1903–1987) in his monograph with the impressive German title *Grundbegriffe der Wahrscheinlichkeitsrechnung* (“Foundations of Probability Theory”). The **Kolmogorov axioms** are simple and intuitive:

1. Once you identify all elementary outcomes, or events, you may then assign *probabilities* to them, where...
2. ... a *probability* is a number between 0 and 1, and an event which is certain has probability 1.
3. Finally, the probability of any event can be calculated using a deceptively simple rule — the **additivity axiom**: *whenever an event can occur in several mutually exclusive ways, the probability for the event is the sum of the probabilities for each way considered separately.*

Obvious, isn't it? So obvious, in fact, that probability theory was accepted as a mathematical framework, a language that can be used to describe actual physical phenomena. Physics should be able to identify elementary events and assign numerical probabilities to them. Once this is done we may revert to mathematical formalism of probability theory. The Kolmogorov axioms will take care of the mathematical consistency and will guide us whenever there is a need to calculate probabilities of more complex events. This is a very sensible approach, apart from the important fact that *it does not always work!* Today, we know that probability theory, as ubiquitous as it is, fails to describe many common quantum phenomena. In order to see the need for quantum theory let us consider a simple experiment in which probability theory fails to give the right predictions.

1.3 The double-slit experiment

In a double-slit experiment, a particle (such as a photon) emitted from a source S can reach a detector D by taking two different paths, e.g. through an upper or a lower slit in a barrier between the source and the detector. After sufficiently many repetitions of this experiment we can evaluate the frequency of clicks in the detector D and show

We will, however, amend the two rules later on when we touch upon particle statistics.

It's an interesting coincidence that the two basic ingredients of modern quantum theory — probability and complex numbers — were discovered by the same person, an extraordinary man of many talents: a gambling scholar by the name of [Girolamo Cardano](#) (1501–1576).

that it is inconsistent with the predictions based on probability theory. Let us use the quantum approach to show how the discrepancy arises.

The particle emitted from S can reach detector D by taking two different paths, which are assigned probability amplitudes z_1 and z_2 , respectively. We may then say that the upper slit is taken with probability $p_1 = |z_1|^2$ and the lower slit with probability $p_2 = |z_2|^2$. These are two mutually exclusive events. With the two slits open, allowing the particle to take either path, probability theory declares (by the Kolmogorov additivity axiom) that the particle should reach the detector with probability $p_1 + p_2 = |z_1|^2 + |z_2|^2$. But this is *not* what happens experimentally!

Let us see what happens if we instead follow the two “quantum rules”: first we add the amplitudes, then we square the absolute value of the sum to get the probability. Thus, the particle will reach the detector with probability

$$\begin{aligned} p &= |z|^2 \\ &= |z_1 + z_2|^2 \\ &= |z_1|^2 + |z_2|^2 + z_1^* z_2 + z_1 z_2^* \\ &= p_1 + p_2 + |z_1||z_2| \left(e^{i(\varphi_2 - \varphi_1)} + e^{-i(\varphi_2 - \varphi_1)} \right) \\ &= p_1 + p_2 + \underbrace{2\sqrt{p_1 p_2} \cos(\varphi_2 - \varphi_1)}_{\text{interference terms}} \end{aligned} \quad (\dagger)$$

where we have expressed the amplitudes in their polar forms:

$$\begin{aligned} z_1 &= |z_1|e^{i\varphi_1} \\ z_2 &= |z_2|e^{i\varphi_2}. \end{aligned}$$

The appearance of the interference terms marks the departure from the classical theory of probability. The probability of any two seemingly mutually exclusive events is the sum of the probabilities of the individual events $p_1 + p_2$ modified by the **interference term** $2\sqrt{p_1 p_2} \cos(\varphi_2 - \varphi_1)$. Depending on the **relative phase** $\varphi_2 - \varphi_1$, the interference term can be either negative (giving what we call **destructive interference**) or positive (**constructive interference**), leading to either suppression or enhancement (respectively) of the total probability p .

The algebra is simple; our focus is on the physical interpretation. Firstly, note that the important quantity here is the *relative phase* $\varphi_2 - \varphi_1$ rather than the individual phases φ_1 and φ_2 . This observation is not trivial at all: if a particle reacts only to the difference of the two phases, each pertaining to a separate path, then it must have, somehow, experienced the two paths, right? That is, we cannot say that the particle has travelled *either* through the upper or the lower slit, because it has travelled through *both*. In the same way, quantum computers follow, in some tangible way, *all* computational paths simultaneously, producing answers that depend on *all* these alternative calculations. Weird, but this is how it is!

Secondly, what has happened to the additivity axiom in probability theory? What was wrong with it? One problem is the assumption that the processes of taking the upper or the lower slit are mutually exclusive; in reality, as we have just mentioned, the two transitions *both occur, simultaneously*. However, we cannot learn this from probability theory, nor from any other *a priori* mathematical construct — we can only observe this by repeated scientific experiments in our physical world.

There is no fundamental reason why Nature should conform to the additivity axiom.

We find out how nature works by making “intelligent” guesses, running experiments, checking what happens and formulating physical theories. If our guess disagrees with experiments then it is wrong, so we try another intelligent guess, and

That is, if one happens then the other one cannot. For example, “heads” and “tails” are mutually exclusive outcomes of flipping a coin, but “heads” and “6” are *not* mutually exclusive outcomes of simultaneously flipping a coin and rolling a dice.

According to the philosopher [Karl Popper](#) (1902–1994) a theory is genuinely scientific only if it is possible, in principle, to establish that it is false. Genuinely scientific theories are never finally confirmed because, no matter how many confirming observations have been made, observations that are inconsistent with the empirical predictions of the theory are always possible.

another, etc. Right now, quantum theory is the best guess we have: it offers good explanations and predictions that have not been falsified by any of the existing experiments. This said, rest assured that one day quantum theory *will* be falsified, and then we will have to start guessing all over again.

The quantum eraser.

This section is not yet finished.

1.4 Superpositions

Amplitudes are more than just tools for calculating probabilities: they tell us something about physical reality. When we deal with probabilities, we may think about them as numbers that quantify our lack of knowledge. Indeed, classically, when we say that “a particle goes through the upper or the lower slit with some respective probabilities”, what we really mean is that it does go through *one* of the two slits, but we just do not know which one for sure. In contrast, according to quantum theory, a particle that goes through the upper and the lower slit with certain amplitudes does explore *both* of the two paths, not just one of them. This is a statement about a real physical situation — about something that is out there and with which we can experiment.

The assumption that the particle goes through one of the two slits but we just don't know which one, is inconsistent with *many* experimental observations.

We have to accept that, apart from some easy to visualise states, known as the **basis states** (such as the particle at the upper slit or the particle at the lower slit), there are infinitely many other states, all of them equally real, in which the particle is in a **superposition** of the two basis states. This rather bizarre picture of reality is the best we have at the moment, and it works (at least, for now!).

Physicists write such superposition states as

$$|\psi\rangle = \alpha|{\text{upper slit}}\rangle + \beta|{\text{lower slit}}\rangle,$$

meaning the particle goes through the upper slit with amplitude α , and through the lower slit with amplitude β . Mathematically, you can think about this expression as a vector $|\psi\rangle$ in a two-dimensional complex vector space written in terms of the two basis vectors $|{\text{upper slit}}\rangle$ and $|{\text{lower slit}}\rangle$. You could also write this vector as a column vector with two complex entries α and β , but then you would have to explain the *physical meaning* of the basis states. Here, we use the **Dirac notation** $|\ \rangle$, introduced by [Paul Dirac](#) (1902–1984) in the early days of the quantum theory as a useful way to write and manipulate vectors. In Dirac notation you can put into the “box” $|\ \rangle$ anything that serves to specify what the vector is: it could be $|\uparrow\rangle$ for spin up and $|\downarrow\rangle$ for spin down (whatever this technical terminology “spin” means), or $|0\rangle$ for a quantum bit holding logical 0 and $|1\rangle$ for a quantum bit holding logical 1, etc. As we shall soon see, there is much more to this notation, and learning to manipulate it will help you greatly.

Dirac notation will likely be familiar to physicists, but may look odd to mathematicians or computer scientists. Love it or hate it (and we suggest the former), the notation is so common that you simply have no choice but to learn it, especially if you want to study anything related to quantum theory.

1.5 Interferometers

One of the most fundamental family of experiments for our purposes are so-called **interference experiments**, modern versions of which are performed using internal degrees of freedom of atoms and ions. For example, **Ramsey interferometry**, named

after American physicist [Norman Ramsey](#) (1915–2011), is a generic name for an interference experiment in which atoms are sent through two separate “resonant interaction” zones, known as **Ramsey zones**, separated by an intermediate “dispersive interaction” zone.

Many beautiful experiments of this type were carried out in the 1990s in [Serge Haroche](#)'s lab at the Ecole Normale Supérieure in Paris. Rubidium atoms were sent through two separate interaction zones (resonant interaction in the first and the third cavity) separated by a phase inducing dispersive interaction zone (the central cavity). The atoms were subsequently measured, via a selective ionisation, and found to be in one of the two preselected energy states, here labeled as $|0\rangle$ and $|1\rangle$. The fraction of atoms found in states $|0\rangle$ or $|1\rangle$ showed a clear dependence on the phase shifts induced by the dispersive interaction in the central cavity. In 2012, Serge Haroche and Dave Wineland shared the Nobel Prize in physics for “ground-breaking experimental methods that enable measuring and manipulation of individual quantum systems.” Let us now try to understand what this experiment actually entails.

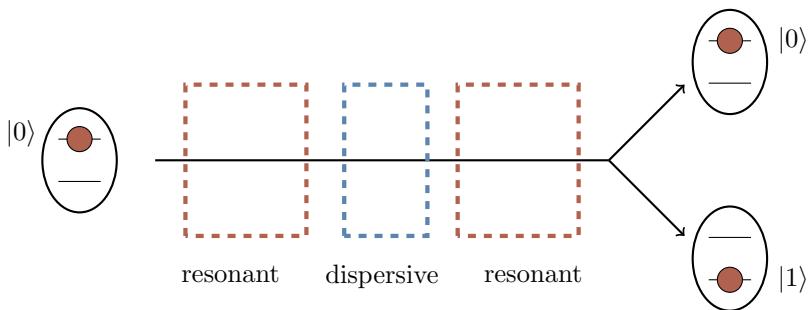


Figure 1.1: A schematic diagram of a Ramsey interference experiment.

The three rectangular boxes in Figure 1.1 represent three cavities, each cavity being an arrangement of mirrors which traps electromagnetic field (think about standing waves in between two mirrors). The oval shapes represent rubidium atoms with two preselected energy states labelled as $|0\rangle$ and $|1\rangle$. Each atom is initially prepared in a highly excited internal energy state $|0\rangle$ and zips through the three cavities, from the left to the right. In each cavity the atom interacts with the cavity field. The first and the third cavities are, for all theoretical purposes, identical: their frequencies are tuned to the resonant frequency of the atom, and the atom exchanges energy with the cavity, going back and forth between its energy states $|0\rangle$ and $|1\rangle$. In contrast, in the second (central) cavity, the atom undergoes the so-called dispersive interaction: it is too off-resonance for the atom to exchange energy with the field, but the atom's energy states “feel” the field and acquire phase shifts. After experiencing this well timed sequence of resonant-dispersive-resonant interactions, the energy of the atom is measured and the atom is found to be either in state $|0\rangle$ or state $|1\rangle$. The (surprising) result of this experiment is analogous to that of the double-slit experiment described above: the fraction of atoms found in state $|0\rangle$ or $|1\rangle$ shows a clear dependence on the phase shifts induced by the dispersive interaction in the central cavity.

We can understand this interference better if we follow the two internal states of the atom as it moves through the three cavities.

If the language of energy states of atoms is unfamiliar to you, don't worry! Here we are just trying to give some physical motivation for one of the fundamental quantum circuits which we will see pop up time and time again. Vaguely, though, you can just have in your mind the idea that atoms have a certain amount of energy at any given time, but the amount that they can have is one of a number of fixed amounts, depending on the chemistry of the atom. This is in fact [one of the key principles](#) in the history of quantum physics, and is the reason for the word “quantum”.

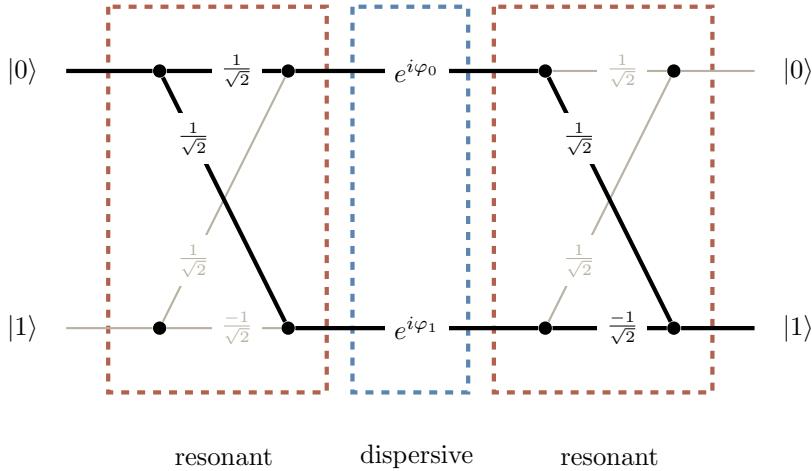


Figure 1.2: The Ramsey interferometer represented as an abstract diagram, to be read from left to right. The line segments represent transitions between the two states, $|0\rangle$ and $|1\rangle$, and the numbers are the corresponding probability amplitudes.

Suppose we are interested in the probability that the atom, initially in state $|0\rangle$, will be found, after completing its journey through the three cavities, in state $|1\rangle$. As you can see in Figure 1.2, this can happen in two ways, as indicated by the two thick paths connecting the input state $|0\rangle$ on the left with the output state $|1\rangle$ on the right. Again, let U_{ij} denote the probability amplitude that input $|j\rangle$ generates output $|i\rangle$ (for $i, j = 0, 1$).

We can see from the diagram that

$$\begin{aligned} U_{10} &= \frac{1}{\sqrt{2}} e^{i\varphi_0} \frac{1}{\sqrt{2}} + \frac{1}{\sqrt{2}} e^{i\varphi_1} \frac{-1}{\sqrt{2}} \\ &= \frac{1}{2} (e^{i\varphi_0} - e^{i\varphi_1}). \end{aligned}$$

Then, using the trick of writing $x = \frac{x+y}{2} + \frac{x-y}{2}$ and $y = \frac{x+y}{2} - \frac{x-y}{2}$, followed by Euler's formula ($e^{i\alpha} = \cos \alpha + i \sin \alpha$), we see that

$$\begin{aligned} U_{10} &= \frac{1}{2} (e^{i\varphi_0} - e^{i\varphi_1}) \\ &= \frac{1}{2} \left(e^{i\frac{\varphi_0+\varphi_1}{2}} e^{i\frac{\varphi_0-\varphi_1}{2}} - e^{i\frac{\varphi_0+\varphi_1}{2}} e^{-i\frac{\varphi_0-\varphi_1}{2}} \right) \\ &= \frac{1}{2} e^{i\frac{\varphi_0+\varphi_1}{2}} \left(e^{i\frac{\varphi_0-\varphi_1}{2}} - e^{-i\frac{\varphi_0-\varphi_1}{2}} \right) \\ &= \frac{1}{2} e^{i\frac{\varphi_0+\varphi_1}{2}} \left(2i \sin \left(\frac{\varphi_0 - \varphi_1}{2} \right) \right) \\ &= -ie^{i\frac{\varphi_0+\varphi_1}{2}} \sin \frac{\varphi_1 - \varphi_0}{2} \end{aligned}$$

where the relative phase $\varphi = \varphi_1 - \varphi_0$ shows up yet again.

The corresponding probability (i.e. that an atom, initially in state $|0\rangle$, will be found in state $|1\rangle$) is then

$$\begin{aligned} P_{10} &= |U_{10}|^2 \\ &= \left| -ie^{i\frac{\varphi_0+\varphi_1}{2}} \sin \frac{\varphi_1 - \varphi_0}{2} \right|^2 \\ &= \left| \sin \frac{\varphi_1 - \varphi_0}{2} \right|^2 \\ &= \frac{1}{2} (1 - \cos \varphi) \end{aligned}$$

From the classical probability theory perspective the resonant interaction induces a random switch between $|0\rangle$ and $|1\rangle$ (why?) and the dispersive interaction has no effect on these two states (why?). One random switch followed by another random switch is exactly the same as a single random switch (if you flip a coin twice and just observe the last result, this is probabilistically the same as just flipping the coin once), which gives $\frac{1}{2}$ for the probability that input $|0\rangle$ becomes output $|1\rangle$.

where we use the fact that $|i|^2 = 1$ and $|e^{i\alpha}|^2 = 1$ for any α , along with the [double angle formula](#) $\cos 2\theta = 1 - 2 \sin^2 \theta$.

You should recognise the first term $\frac{1}{2}$ as the “classical” probability and the second one $-\frac{1}{2} \cos \varphi$ as the interference term. We can repeat such calculations for any other pair of input–output states. This approach works fine here but, in general, tracking all possible paths in evolving quantum systems can become messy when the number of input and output states increases. There is, however, a neat way of doing these calculations: matrix multiplication.

The effect of each interaction on atomic states can be described by a matrix of transition amplitudes, as illustrated in Figure 1.3, and then the sequence of independent interactions is described by the product of these matrices: we compile all the U_{ij} into one matrix U .

$$\begin{aligned} U &= \begin{bmatrix} \frac{1}{\sqrt{2}} & \frac{1}{\sqrt{2}} \\ \frac{1}{\sqrt{2}} & \frac{-1}{\sqrt{2}} \end{bmatrix} \begin{bmatrix} e^{i\varphi_0} & 0 \\ 0 & e^{i\varphi_1} \end{bmatrix} \begin{bmatrix} \frac{1}{\sqrt{2}} & \frac{1}{\sqrt{2}} \\ \frac{1}{\sqrt{2}} & \frac{-1}{\sqrt{2}} \end{bmatrix} \\ &= e^{i\frac{\varphi_0+\varphi_1}{2}} \begin{bmatrix} \cos \frac{\varphi}{2} & -i \sin \frac{\varphi}{2} \\ -i \sin \frac{\varphi}{2} & \cos \frac{\varphi}{2} \end{bmatrix} \\ &= \begin{bmatrix} U_{00} & U_{01} \\ U_{10} & U_{11} \end{bmatrix} \end{aligned}$$

where $\varphi = \varphi_1 - \varphi_0$, as before.

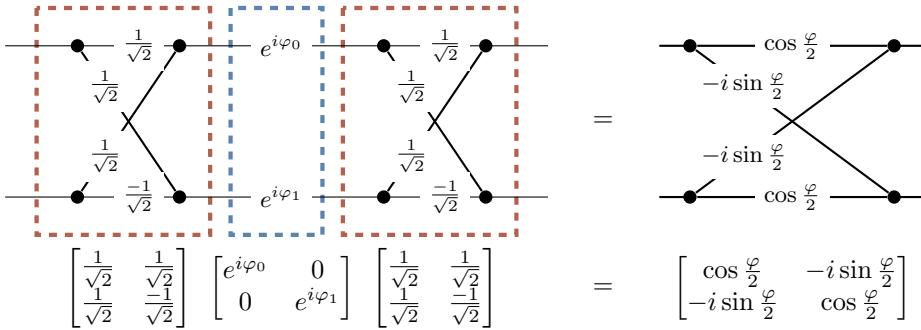


Figure 1.3: The Ramsey interferometer represented as an abstract diagram (matrix approach). Here we have omitted the $|0\rangle$ and $|1\rangle$ labels, just to simplify the diagram. We also ignore (for reasons that we will later explain) the **global** phase factor of $e^{i\frac{\varphi_0+\varphi_1}{2}}$.

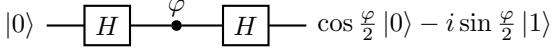
In general, quantum operation A followed by quantum operation B is the quantum operation described by the matrix product BA . Indeed, the expression $(BA)_{ij} = \sum_k B_{ik} A_{kj}$ is the sum over amplitudes that input $|j\rangle$ generates output $|i\rangle$ via a specific intermediate state $|k\rangle$. As you can see, the matrix approach is a wonderful bookkeeping tool: in one package it takes care of both multiplying *and* adding probability amplitudes corresponding to all the contributing paths.

Note the order of the matrices: the composition “ A followed by B ” is BA , not AB ! This reflects the fact that, in linear algebra, we apply a matrix to a vector on the left, i.e. A applied to v is written Av .

1.6 Qubits, gates, and circuits

Atoms, trapped ions, molecules, nuclear spins, and many other quantum objects with two pre-selected basis states labelled as $|0\rangle$ and $|1\rangle$ can be used to implement simple quantum interference — from now on we will call such objects **quantum bits**, or **qubits**. But note that there is no need to learn about the physics behind these diverse technologies (e.g. “what is nuclear spin?”) if all you want is to understand the basics of quantum theory. Indeed, from now on we will conveniently forget about any specific experimental realisation of a qubit and represent a generic **single-qubit interference** graphically as a **circuit diagram**:

Do not confuse the interference diagrams of Figure 1.1 and Figure 1.3 with the circuit diagrams. In the circuit diagrams, which we will use almost constantly from now on, a single line represents a single qubit.



This diagram should be read *from left to right*. The horizontal line represents a qubit that is inertly carried from one quantum operation to another. We often call this line a **quantum wire**. The wire may describe translation in space (e.g. atoms travelling through cavities) or translation in time (e.g. a sequence of operations performed on a trapped ion). The boxes or circles on the wire represent elementary quantum operations, called **quantum (logic) gates**.

In the example circuit above, we have two types of gates: two **Hadamard gates** H (think “resonant interaction”) and one **phase gate** P_φ (think “dispersive interaction”), where

$$H = \begin{bmatrix} \frac{1}{\sqrt{2}} & \frac{1}{\sqrt{2}} \\ \frac{1}{\sqrt{2}} & \frac{-1}{\sqrt{2}} \end{bmatrix} \quad \text{and} \quad P_\varphi = \begin{bmatrix} 1 & 0 \\ 0 & e^{i\varphi} \end{bmatrix}.$$

The input qubits appear as state vectors on the left side of circuit diagrams, and the output qubits as state vectors on the right. The product of the three matrices

$$HP_\varphi H = \begin{bmatrix} \cos \frac{\varphi}{2} & -i \sin \frac{\varphi}{2} \\ -i \sin \frac{\varphi}{2} & \cos \frac{\varphi}{2} \end{bmatrix}$$

describes the action of the whole circuit, telling us that it maps input state vectors to output state vectors as follows:

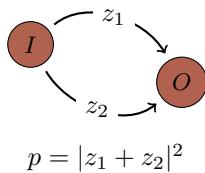
$$\begin{aligned} |0\rangle &\mapsto \cos \frac{\varphi}{2} |0\rangle - i \sin \frac{\varphi}{2} |1\rangle, \\ |1\rangle &\mapsto -i \sin \frac{\varphi}{2} |0\rangle + \cos \frac{\varphi}{2} |1\rangle. \end{aligned}$$

1.7 Quantum decoherence

We do need quantum theory to describe many physical phenomena, but, at the same time, there are many other phenomena where the classical theory of probability works pretty well. Indeed, we hardly see quantum interference on a daily basis. Why? The answer is **decoherence**. The addition of probability amplitudes, rather than probabilities, applies to physical systems which are *completely isolated*. However, it is almost impossible to isolate a complex quantum system, such as a quantum computer, from the rest of the world: there will always be spurious interactions with the environment (such as heat transfer), and when we add amplitudes, we have to take into account not only different configurations of the physical system at hand, but also different configurations of the environment.

For example, consider an isolated system composed of a quantum computer and its environment. The computer is prepared in some input state I and generates output O . Let us look at the following two scenarios:

1. *The computer is isolated and quantum computation does not affect the environment.* The computer and the environment evolve independently from each other and, as a result, the environment does not hold any physical record of how the computer reached output O . In this case we add the amplitudes for each of the two alternative computational paths.

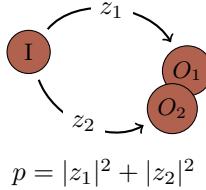


Yet again we see that we can conveniently forget about the actual physical implementations, treating them all with the same abstract description.

Global phase factors are irrelevant; it is only the *relative* phase $\varphi = \varphi_1 - \varphi_0$ that matters. For example, in a single-qubit phase gate P_φ we usually factor out $e^{i\varphi_0}$, leaving us with the two diagonal entries: 1 and $e^{i\varphi}$.

cf. Figure 1.3. Note again that circuits are read *left to right*, but matrix composition goes *right to left*. Since the first and last matrices/gates are the same (i.e. both are H), we don't notice this, but it's important to note that the *first* (i.e. leftmost) H in the matrix product $HP_\varphi H$ corresponds to the *last* (i.e. rightmost) H in the circuit diagram.

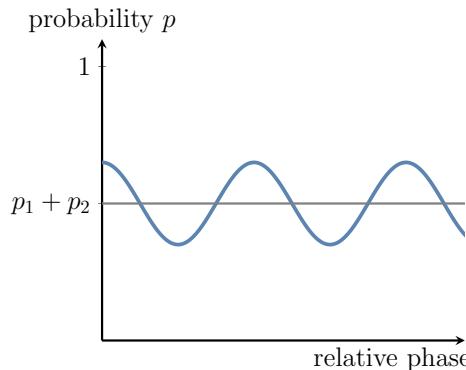
2. *Quantum computation affects the environment.* The environment now holds a physical record of how the computer reached output O , which results in two final states of the composed system (computer *and* environment) which we denote O_1 and O_2 . We add the probabilities for each of the two alternative computational paths.



When quantum computation affects the environment (or vice versa), we have to include the environment in our analysis, since it is now involved in the computation. Depending on which computational path was taken, the environment may end up in two distinct states. The computer itself may show output O , but when we include the environment we have not one, but two, outputs, O_1 and O_2 , denoting, respectively, “computer shows output O and the environment knows that path 1 was taken” and “computer shows output O and the environment knows that path 2 was taken”. There are no alternative ways of reaching O_1 or O_2 , hence there is no interference, and the corresponding probabilities read $p_1 = |z_1|^2$ for O_1 , and $p_2 = |z_2|^2$ for O_2 . The probability that the computer shows output O , regardless the state of the environment, is the sum of the two probabilities: $p = p_1 + p_2$. We have lost the interference term and, with this, any advantages of quantum computation are also lost. In the presence of decoherence, the interference formula in [Equation \(‡\)](#) is modified and reads

$$p = p_1 + p_2 + 2v\sqrt{p_1 p_2} \cos(\varphi_2 - \varphi_1),$$

where the parameter v , called the **visibility** of the interference pattern, ranges from 0 (the environment can perfectly distinguish between the two paths, i.e. **total decoherence**, or **no interference**) to 1 (the environment cannot distinguish between the two paths, i.e. **no decoherence**, or **full interference**), with the values in between corresponding to **partial decoherence**.



We shall derive this formula later on, and you will see that v quantifies the degree of distinguishability between O_1 and O_2 . The more the environment knows about which path was taken, the less interference we see, and the less we can leverage the computational power of quantum effects.

Decoherence suppresses quantum interference.

Decoherence is chiefly responsible for our classical description of the world: without interference terms we may as well add probabilities instead of amplitudes (thus recovering the additivity axiom). While decoherence is a serious impediment to building quantum computers, depriving us of the power of quantum interference, it is not all doom and gloom: there are clever ways around decoherence, such as quantum error correction and fault-tolerant methods, both of which we will meet later.

1.8 Types of computation

One single qubit has two logical (i.e. non-superposition) states: $|0\rangle$ and $|1\rangle$. Bring another qubit and the combined systems has four logical states: $|00\rangle$, $|01\rangle$, $|10\rangle$, and $|11\rangle$. In general n qubits will give us 2^n states, representing all possible binary strings of length n . It is important to use subsystems — here qubits — rather than one chunk of matter, since, by operating on at most n qubits, we can reach any of the 2^n states of the composed system. Now, if we let the qubits interact in a controllable fashion, then we are computing!

Think about computation as a physical process that evolves a prescribed initial configuration of a computing machine, called **INPUT**, into some final configuration, called **OUTPUT**. We shall refer to the configurations as **states**. Figure 1.4 shows five consecutive computational steps performed on four distinct states.

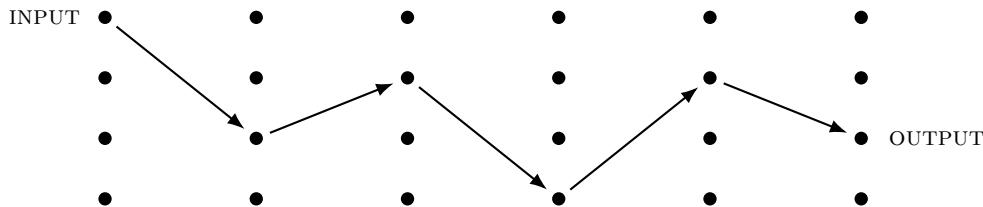


Figure 1.4: Deterministic computation.

That computation was **deterministic**: every time you run it with the same input, you get the same output.

But a computation does not have to be deterministic — we can augment a computing machine by allowing it to “toss an unbiased coin” and to choose its steps randomly. It can then be viewed as a directed tree-like graph where each node corresponds to a state of the machine, and each edge represents one step of the computation, as shown in Figure 1.5

So we read left to right, and omit the arrowheads.

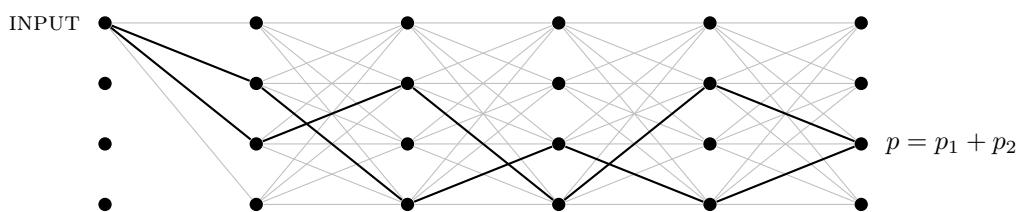


Figure 1.5: Probabilistic computation.

The computation starts from some initial state (**INPUT**) and it subsequently branches into other nodes representing states reachable with non-zero probability from the initial state. The probability of a particular final state (**OUTPUT**) being reached is equal to the sum of the probabilities along all mutually exclusive paths which connect the initial state with that particular state. Figure 1.5 shows only two computational paths, but, in general, there could be many more paths (here, up to 256) contributing to the final probability.

Quantum computation can be represented by a similar graph, as in Figure 1.6.

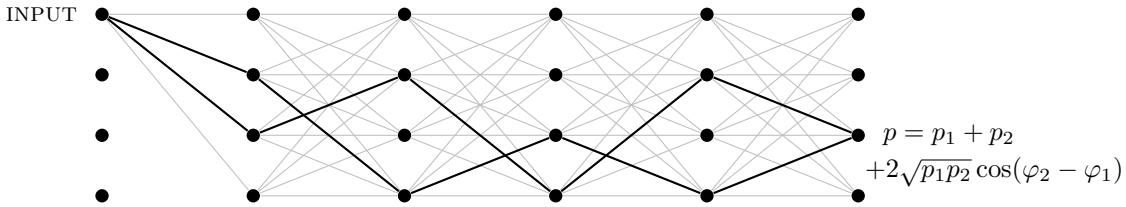


Figure 1.6: Quantum computation.

For quantum computations, we associate with each edge in the graph the probability *amplitude* that the computation follows that edge. The probability amplitude that a particular path to be followed is the product of amplitudes pertaining to the transitions in each step. The probability amplitude of a particular final state being reached is equal to the sum of the amplitudes along all mutually exclusive paths which connect the initial state with that particular state:

$$z = \sum_{\text{all paths } k} z_k.$$

The resulting probability, as we have just seen, is the sum of the probabilities p_k pertaining to each computational path modified by the interference terms. To show this, note first that

$$\begin{aligned} p &= |z|^2 = z^* z \\ &= \left(\sum_{j=1}^N z_j \right)^* \left(\sum_{k=1}^N z_k \right) \\ &= (z_1^* + z_2^* + \dots + z_N^*) (z_1 + z_2 + \dots + z_N) \end{aligned}$$

Multiplying out these two sums gives us terms of the form $z_i^* z_j$ for $1 \leq i, j \leq N$, so we can think of these as forming a square matrix and then split the sum into the “diagonal” terms and the “off-diagonal” terms:

$$\begin{aligned} p &= \underbrace{\sum_k |z_k|^2}_{\text{diagonal elements}} + \underbrace{\sum_{k>j} (z_k^* z_j + z_j^* z_k)}_{\text{off-diagonal elements}} \\ &= \sum_k |z_k|^2 + \sum_{k>j} (|z_k| |z_j| e^{i(\varphi_j - \varphi_k)} + |z_j| |z_k| e^{i(\varphi_k - \varphi_j)}) \\ &= \sum_k |z_k|^2 + \sum_{k>j} (|z_k| |z_j| e^{-i(\varphi_k - \varphi_j)} + |z_j| |z_k| e^{i(\varphi_k - \varphi_j)}) \\ &= \sum_k p_k + \sum_{k>j} 2|z_k||z_j| \cos(\varphi_k - \varphi_j) \\ &= \sum_k p_k + \sum_{k>j} \underbrace{2\sqrt{p_k p_j} \cos(\varphi_k - \varphi_j)}_{\text{interference terms}}. \end{aligned}$$

Quantum computation can be viewed as a complex multi-particle quantum interference involving many computational paths through a computing device. The art of quantum computation is to *shape* the quantum interference through a sequence of computational steps, enhancing probabilities of the “correct” outputs and suppressing probabilities of the “wrong” ones.

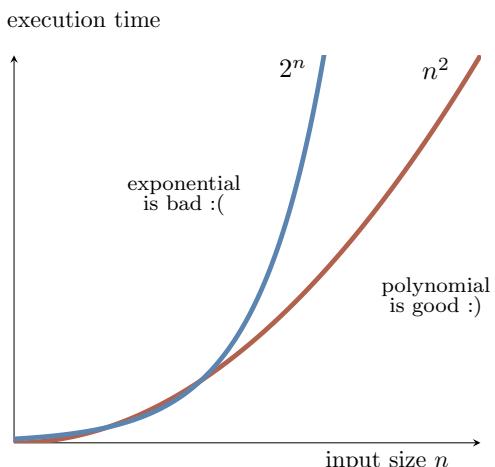
1.9 Computational complexity

Is there a compelling reason why we should care about quantum computation? It may sound like an extravagant way to compute something that can be computed anyway. Indeed, your standard laptop, given enough time and memory, can simulate pretty much any physical process. In principle, it can also simulate any quantum interference and compute everything that quantum computers can compute. The snag is that this simulation is, in general, very inefficient. And efficiency does matter, especially if you have to wait more than the age of the universe for your laptop to stop and deliver an answer!

In order to solve a particular problem, computers (classical or quantum) follow a precise set of instructions called an **algorithm**. Computer scientists quantify the efficiency of an algorithm according to how rapidly its running time, or the use of memory, increases when it is given ever larger inputs to work on. An algorithm is said to be **efficient** if the number of elementary operations taken to execute it increases no faster than a polynomial function of the size of the input. We take the input size to be the total number of binary digits (bits) needed to specify the input. For example, using the algorithm taught in elementary school, one can multiply two n digit numbers in a time that grows like the number of digits squared, n^2 . In contrast, the fastest-known method for the reverse operation — factoring an n -digit integer into prime numbers — takes a time that grows exponentially, roughly as 2^n . This is considered inefficient.

The age of the universe is currently estimated to be around 13.772 billion years.

Note that technological progress alone, such as increasing the speed of classical computers, will never turn an inefficient (exponential scaling) algorithm into an efficient (polynomial scaling) algorithm. Why?

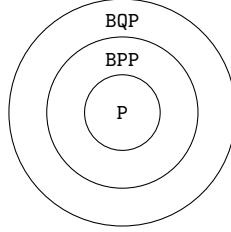


The class of problems that can be solved in polynomial time by a deterministic computer is represented by the capital letter P, for **polynomial** time. The class of problems that can be solved in polynomial time by a probabilistic computer is called BPP, for **bounded-error probabilistic polynomial** time.

It is clear that BPP contains P, since deterministic computation is a special case of probabilistic computation in which we never consult the source of randomness. When we run a probabilistic (a.k.a. randomised) computation many times on the same input, we will not get the same answer every time, but the computation is useful if the probability of getting the right answer is high enough.

Finally, the class of problems that can be solved in polynomial time by a quantum computer is called BQP, for *bounded-error quantum polynomial*. Since a quantum computer can easily generate random bits and simulate a probabilistic classical computer, BQP certainly contains the class BPP (which itself contains the class P). Here we are interested in problems that are in BQP but *not known to be* in BPP. The most popular example of such a problem is factoring.

The phrase “bounded-error” has a precise meaning: the probability of error is at most 1/3.



A quantum algorithm, discovered by Peter Shor in 1994, can factor n -digit numbers in a number of steps that grows only as n^2 , as opposed to the 2^n that we have classically. Since the intractability of factorisation underpins the security of many methods of encryption, Shor's algorithm (see Section 10.11) was soon hailed as the first “killer application” for quantum computation: something very useful that only a quantum computer could do. Since then, the hunt has been on for interesting things for quantum computers to do, and at the same time, for the scientific and technological advances that could allow us to build quantum computers in reality.

It must be stressed that *not all* quantum algorithms are so efficient. In fact many are no faster than their classical counterparts. Which particular problems will lend themselves to quantum speed-ups is an open question.

1.10 Outlook

When the physics of computation was first investigated, starting in the 1960s, one of the main motivations was a fear that quantum-mechanical effects might place fundamental bounds on the accuracy with which physical objects could render the properties of the abstract entities (such as logical variables and operations) that appear in the theory of computation. It turned out, however, that quantum mechanics itself imposes no significant limits, and even breaks through some of those problems that were imposed by *classical* physics. The quantum world has a richness and intricacy that allows new practical technologies, and new kinds of knowledge. In this course we will merely scratch the surface of the rapidly developing field of quantum computation. We will concentrate mostly on the fundamental issues and skip many experimental details. However, it should be mentioned that quantum computing is a serious possibility for future generations of computing devices. At present it is not clear how and when fully-fledged quantum computers will eventually be built, but this notwithstanding, the quantum theory of computation already plays a much more fundamental role in the scheme of things than its classical predecessor did. It is reasonable to argue that anyone who seeks a fundamental understanding of either physics, computation, or logic must incorporate into their world view the new insights brought by quantum theory.

1.11 Remarks and exercises

1.11.1 A historical remark

Back in 1926, Max Born simply postulated the connection between amplitudes and probabilities, but did not get it quite right on his first attempt. In the original paper proposing the probability interpretation of the state vector (wavefunction) he wrote:

... If one translates this result into terms of particles only one interpretation is possible. $\Theta_{\eta,\tau,m}(\alpha, \beta, \gamma)$ [the wavefunction for the particular problem he is considering] gives the probability* for the electron arriving from the z direction to be thrown out into the direction designated by the angles $\alpha, \beta, \gamma \dots$

* Addition in proof: More careful considerations show that the probability is proportional to the square of the quantity $\Theta_{\eta,\tau,m}(\alpha, \beta, \gamma)$.

Max Born, “Zur Quantenmechanik der Stoßvorgänge”, *Zeitschrift für Physik* 37 (1926), pp. 893–867.

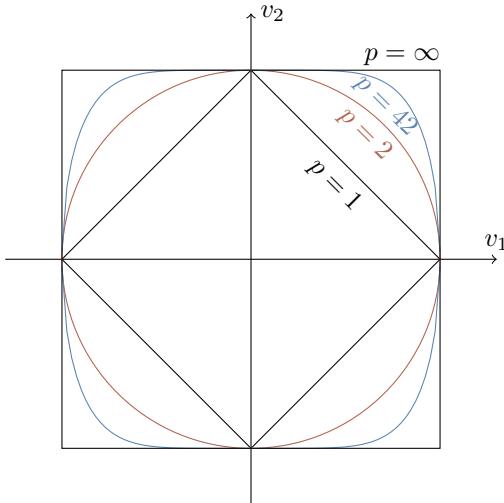
1.11.2 Modifying the Born rule

Suppose that we modified the Born rule, so that probabilities were given by the absolute values of amplitudes *raised to the power p* (for some $p > 0$ not necessarily equal to 2). Then physically admissible evolutions would still have to preserve the normalisation of probability: mathematically speaking, they would have to be *isometries of p-norms*.

The p -norm of a vector $v = (v_1, v_2, \dots, v_n)$, for $p \in \mathbb{N}$, is defined as

$$\sqrt[p]{|v_1|^p + |v_2|^p + \dots + |v_n|^p}.$$

It is clear that any permutation of vector components and multiplication by phase factors (i.e. unit complex numbers, of the form $e^{i\varphi}$ for some φ) will leave any p -norm unchanged. It turns out that these complex permutations are the *only* isometries, except for *one* special case: $p = 2$. For $p = 2$, the isometries are exactly unitaries, which form a *continuous* group; in all other cases we are restricted to discrete permutations. We do not have to go into details of the proof since we can *see* this result.



In the case $p = 2$, we recover the usual Pythagorean/Euclidean equation that we all know and love: the distance of the point (v_1, v_2, \dots, v_n) from the origin is $\sqrt{v_1^2 + v_2^2 + \dots + v_n^2}$; if we take $n = 2$ as well then we recover the Pythagoras theorem.

Figure 1.7: The unit spheres in the p -norm for $p = 1, 2, 4, \infty$ (where the definition of the ∞ -norm is slightly different; we will come back to this in Section 12.11.2).

The image of the unit sphere must be preserved under probability preserving operations. As we can see in Figure 1.7, the 2-norm is special because of its rotational invariance (it describes a circle) — the probability measure picks out no preferred basis in the space of state vectors. Moreover, it respects unitary operations and does not restrict them in any way. If the admissible physical evolutions were restricted to discrete symmetries, e.g. permutations, then there would be no continuity, and no concept of “time” as we know it.

1.11.3 Many computational paths

A quantum computer starts calculations in some initial state, then follows n different computational paths which lead to the final output. The computational paths are followed with probability amplitudes $\frac{1}{n}e^{ik\varphi}$, where φ is a fixed angle $0 < \varphi < 2\pi$ and $k = 0, 1, \dots, n - 1$. Using the fact that $1 + z + z^2 + \dots + z^n = \frac{1-z^{n+1}}{1-z}$, show that the probability P of generating the output is given by

$$P = \frac{1}{n^2} \left| \frac{1-e^{in\varphi}}{1-e^{i\varphi}} \right|^2 = \frac{1}{n^2} \frac{\sin^2(n\frac{\varphi}{2})}{\sin^2(\frac{\varphi}{2})}.$$

for $0 < \varphi < 2\pi$, and that $P = 1$ when $\varphi = 0$. Plot the probability as a function of φ .

1.11.4 Distant photon emitters

Imagine two distant stars, A and B, that emit *identical* photons. If you point a single detector towards them you will register a click every now and then, but you never know which star the photon came from. Now prepare two detectors and point them towards the stars. Assume the photons arrive with the probability amplitudes specified in Figure 1.8. Every now and then you will register a coincidence: the two detectors will both click.

- Calculate the probability of such a coincidence.
- Now assume that $z \approx \frac{1}{r}e^{i2r\pi/\lambda}$, where r is the distance between detectors and the stars and λ is some fixed constant. How can we use this to measure r ?

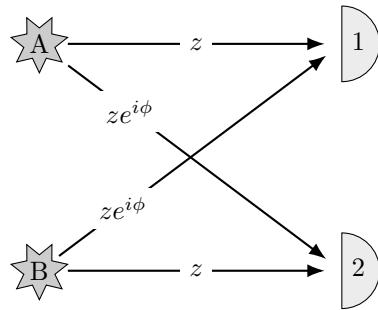


Figure 1.8: Two photon detectors pointing at two stars, with the probabilities of detection labelling the arrows.

1.11.5 Quantum Turing machines

The classical theory of computation is essentially the theory of the [universal Turing machine](#) — the most popular mathematical model of classical computation. Its significance relies on the fact that, given a possibly very large *but still finite* amount of time, the universal Turing machine is capable of any computation that can be done by any modern classical digital computer, no matter how powerful. The concept of Turing machines may be modified to incorporate quantum computation, but we will not follow this path. It is much easier to explain the essence of quantum computation talking about quantum logic gates and quantum Boolean networks or circuits. The two approaches are computationally equivalent, even though certain theoretical concepts, e.g. in computational complexity, are easier to formulate precisely using the Turing machine model. The main advantage of quantum circuits is that they relate far more directly to proposed experimental realisations of quantum computation.

1.11.6 More time, more memory

A quantum machine has N perfectly distinguishable configurations. What is the maximum number of computational paths connecting a specific input with a specific output after k steps of the machine?

Suppose you are using your laptop to add together amplitudes pertaining to each of the paths. As k and N increase you may need more time and more memory to complete the task. How does the execution time and the memory requirements grow with k and N ? In particular, which will be the thing that limits you sooner: not having enough memory, not having enough time, or both?

1.11.7 Asymptotic behaviour: big-O

In order to make qualitative distinctions between how different functions grow we will often use the asymptotic big- \mathcal{O} notation. For example, suppose an algorithm

running on input of size n takes $an^2 + bn + c$ elementary steps, for some positive constants a, b and c . These constants depend mainly on the details of the implementation and the choice of elementary steps. What we really care about is that, for large n , the whole expression is dominated by its quadratic term. We then say that the running time of this algorithm grows as n^2 , and we write it as $\mathcal{O}(n^2)$, ignoring the less significant terms and the constant coefficients. More precisely, let $f(n)$ and $g(n)$ be functions from positive integers to positive reals. You may think of $f(n)$ and $g(n)$ as the running times of two algorithms on inputs of size n . We say $f = \mathcal{O}(g)$, which means that f grows no faster than g , if there is a constant $c > 0$ such that $f(n) \leq cg(n)$ for all sufficiently large values of n . Essentially, $f = \mathcal{O}(g)$ is a very loose analogue of $f \leq g$. In addition to the big- \mathcal{O} notation, computer scientists often use Ω for lower bounds: $f = \Omega(g)$ means $g = \mathcal{O}(f)$. Again, this is a very loose analogue of $f \geq g$.

$f = \mathcal{O}(g)$ is pronounced as “ f is big-oh of g ”.

1. When we say that $f(n) = \mathcal{O}(\log n)$, why don't we have to specify the base of the logarithm?
2. Let $f(n) = 5n^3 + 1000n + 50$. Is $f(n) = \mathcal{O}(n^3)$, or $\mathcal{O}(n^4)$, or both?
3. Which of the following statements are true?
 - a. $n^k = \mathcal{O}(2^n)$ for any constant k
 - b. $n! = \mathcal{O}(n^n)$
 - c. if $f_1 = \mathcal{O}(g)$ and $f_2 = \mathcal{O}(g)$, then $f_1 + f_2 = \mathcal{O}(g)$.

1.11.8 Polynomial is good, and exponential is bad

In computational complexity the basic distinction is between polynomial and exponential algorithms. Polynomial growth is good and exponential growth is bad, especially if you have to pay for it. There is an old story about the legendary inventor of chess who asked the Persian king to be paid only by a grain of cereal, doubled on each of the 64 squares of a chess board. The king placed one grain of rice on the first square, two on the second, four on the third, and he was supposed to keep on doubling until the board was full. The last square would then have $2^{63} = 9,223,372,036,854,775,808$ grains of rice, more than has been ever harvested on planet Earth, to which we must add the grains of all previous squares, making the total number about twice as large. If we placed that many grains in an unbroken line we would reach the nearest star Alpha Centauri, our closest celestial neighbour beyond the solar system, about 4.4 light-years away.

The moral of the story: if whatever you do requires an exponential use of resources, you are in trouble.

One light year (the distance that light travels through a vacuum in one year) is 9.4607×10^{15} metres.

1.11.9 Imperfect prime tester

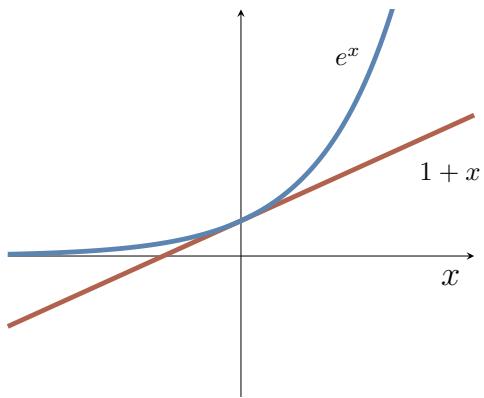
There exists a randomised algorithm which tests whether a given number N is prime. The algorithm always returns yes when N is prime, and the probability it returns yes when N is not prime is ε , where ε is never greater than a half (independently, each time you run the algorithm). You run this algorithm r times (for the same value of N), and each time the algorithm returns yes. What is the probability that N is *not* prime?

Primality used to be given as the classic example of a problem in BPP but not P. However, in 2002 a deterministic polynomial time test for primality was proposed by Manindra Agrawal, Neeraj Kayal, and Nitin Saxena in the wonderfully titled paper “PRIMES is in P”. Thus, since 2002, primality has been in P.

1.11.10 Imperfect decision maker

Suppose a randomised algorithm solves a decision problem, returning yes or no answers. It gets the answer wrong with a probability not greater than $\frac{1}{2} - \delta$, where $\delta > 0$ is a constant. If we are willing to accept a probability of error no larger than ε , then it suffices to run the computation r times, where $r = \mathcal{O}(\log 1/\varepsilon)$.

This result is known as the Chernoff bound.



1. If we perform this computation r times, how many possible sequences of outcomes are there?
2. Give a bound on the probability of any particular sequence with w wrong answers.
3. If we look at the set of r outcomes, we will determine the final outcome by performing a majority vote. This can only go wrong if $w > r/2$. Give an upper bound on the probability of any single sequence that would lead us to the wrong conclusion.
4. Using the bound $1 - x \leq e^{-x}$, conclude that the probability of our coming to the wrong conclusion is upper bounded by $e^{-2r\delta^2}$.

2 Qubits

About quantum bits and quantum circuits, including the “impossible” square root of NOT, as well as an introduction to single-qubit unitaries and rotations of the Bloch sphere, and the implications concerning universal gates.

When studying classical information theory, one single bit isn't usually the most interesting object to think about — it's either 0 or 1. Yet in the quantum case, just working with one “quantum bit” (which we call a **qubit**) opens up a whole world of interesting mathematics. In fact, **single-qubit interference** is arguably the fundamental building block for quantum computing, and so deserves to be thoroughly investigated and understood.

2.1 Composing quantum operations

In order to understand something in its full complexity it is always good to start with the simplest case. Let us take a closer look at quantum interference in the simplest possible computing machine: the one that has only two distinguishable configurations — two quantum states — which we label as $|0\rangle$ and $|1\rangle$. We prepare the machine in some input state, usually $|0\rangle$, and let it **evolve**: the machine undergoes a prescribed sequence of computational steps, each of which induces transitions between the two “computational states” $|0\rangle$ and $|1\rangle$. The machine then ends in the output state $|\psi\rangle = \alpha_0|0\rangle + \alpha_1|1\rangle$, meaning the two outputs, $|0\rangle$ and $|1\rangle$, are reached with probability amplitudes α_0 and α_1 , respectively. In the process of computation each computational step U (also referred to as an **operation**) sends state $|k\rangle$ to state $|l\rangle$, where $k, l = 0, 1$, but only with some **amplitude** U_{lk} . We write this as

$$|k\rangle \mapsto \sum_l U_{lk} |l\rangle.$$

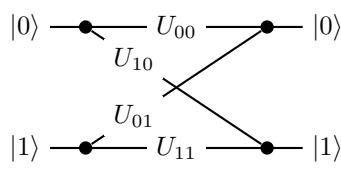
(watch out for the order of the indices). In words, the state $|k\rangle$ evolves into the specific state $|l\rangle$ with probability amplitude U_{lk} and probability $|U_{lk}|^2$, so the whole situation is described by the superposition (i.e. the sum) of all of these.

Thus any computational step U of this machine can be described by a matrix which tabulates all the transition amplitudes:

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

The matrix element U_{lk} represents the amplitude of transition from state $|k\rangle$ to state $|l\rangle$ (again, watch the order of indices). To be clear, the entries in this matrix are not any random complex numbers: their moduli squared represent transition probabilities, which in turn implies that such matrices must be **unitary**.

We can also describe U by drawing a diagram, which contains exactly the same information as the matrix representation, but just in a different form:



Now how can we find some quantum interference to study? Consider two computational steps, U and V . What is the amplitude that input $|k\rangle$ will generate output $|m\rangle$? We have to check all computational paths leading from input $|k\rangle$ to output $|m\rangle$

Recall that matrix U is called **unitary** if

$$U^\dagger U = UU^\dagger = 1$$

where the **adjoint** or **Hermitian conjugate** U^\dagger of any matrix U with complex entries U_{ij} is obtained by taking the complex conjugate of every element in the matrix and then interchanging rows and columns ($U_{kl}^\dagger = U_{lk}^*$).

and add the corresponding amplitudes. For example, as you can see in Figure 2.1, input $|0\rangle$ and output $|1\rangle$ are connected by the two computational paths: $|0\rangle \mapsto |0\rangle \mapsto |1\rangle$ (amplitude $V_{10}U_{00}$) and $|0\rangle \mapsto |1\rangle \mapsto |1\rangle$ (amplitude $V_{11}U_{10}$). Thus the total amplitude that input $|0\rangle$ gives output $|1\rangle$ is the sum $V_{10}U_{00} + V_{11}U_{10}$, and when we take the modulus squared of this expression we will see the interference term.

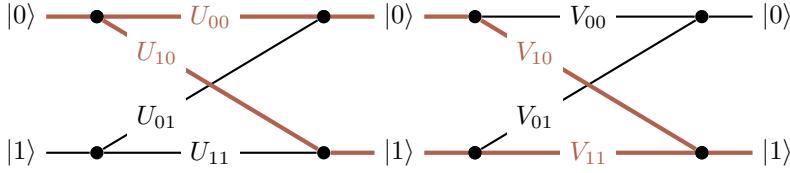


Figure 2.1: The composition of two computational steps, U and V , with the possible paths from $|0\rangle$ to $|1\rangle$ highlighted.

In general, given U and V

$$|k\rangle \mapsto \sum_l U_{lk}|l\rangle$$

$$|l\rangle \mapsto \sum_m V_{ml}|m\rangle$$

we can compose the two operations: we first apply U , and then V , to obtain

$$|k\rangle \mapsto \sum_l U_{lk} \left(\sum_m V_{ml}|m\rangle \right)$$

$$= \sum_m \left(\sum_l V_{ml}U_{lk} \right) |m\rangle$$

$$= \sum_m (VU)_{mk}|m\rangle.$$

If you want to hone your quantum intuition think about it the following way. The amplitude that input $|k\rangle$ evolves to $|m\rangle$ via a specific intermediate state $|l\rangle$ is given by $V_{ml}U_{lk}$ (evolutions are independent so the amplitudes are multiplied). This done, we have to sum over all possible values of l (the transition can occur in several mutually exclusive ways so the amplitudes are added) to obtain $\sum_l V_{ml}U_{lk}$. Thus the matrix multiplication VU (watch the order of matrices) in one swoop takes care of the multiplication *and* addition of amplitudes corresponding to different computational paths.

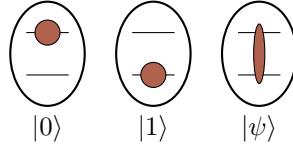
2.2 Quantum bits, called “qubits”

Such a two-state machine that we have just described in abstract terms is usually realised as a controlled evolution of a two-state system, called a **quantum bit**, or **qubit** for short. For example, the state $|0\rangle$ may be chosen to be the lowest energy state of an atom (the **ground state**), and state $|1\rangle$ a higher energy state (the **excited state**). Pulses of light of the appropriate frequency, duration, and intensity can take the atom back and forth between the basis states $|0\rangle$ and $|1\rangle$ (implementing logical NOT).

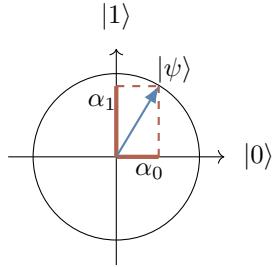
Some other pulses (say, half the duration or intensity) will take the atom into states that have no classical analogue. Such states are called **coherent superpositions** of $|0\rangle$ and $|1\rangle$, and represent a qubit in state $|0\rangle$ with some amplitude α_0 and in state $|1\rangle$ with some other amplitude α_1 . This is conveniently represented by a state vector

$$|\psi\rangle = \alpha_0|0\rangle + \alpha_1|1\rangle \leftrightarrow \begin{bmatrix} \alpha_0 \\ \alpha_1 \end{bmatrix}$$

More general n -state systems can also be of interest, and are sometimes called **q-nits**; three-state systems in particular are sometimes called **qutrits**. In this book, however, we will only concern ourselves with qubits, since they readily generalise the classical notion of bits (and also give us more than enough interesting constructions to get started with!).



By Born's rule, we know that α_0 and α_1 cannot be arbitrary complex numbers: they must satisfy $|\alpha_0|^2 + |\alpha_1|^2 = 1$. This lets us draw the state vector "geometrically", using the fact that the locus of vectors of magnitude equal to 1 describes a circle:



But recall that amplitudes are *complex* numbers, and so α_0 and α_1 cannot really be drawn as 1-dimensional *real* vectors on a flat screen or piece of paper; the picture above provides good intuition, but to be fully accurate we would need to draw it in four-dimensional space (or at least on some three-dimensional paper).

A **qubit** is a quantum system in which the Boolean states 0 and 1 are represented by a prescribed pair of normalised and mutually orthogonal quantum states labelled by $|0\rangle$ and $|1\rangle$. The two states form a so-called **computational** (or **standard**) basis, and so any other state of an isolated qubit can be written as a coherent superposition

$$|\psi\rangle = \alpha_0|0\rangle + \alpha_1|1\rangle$$

for some α_0 and α_1 such that $|\alpha_0|^2 + |\alpha_1|^2 = 1$.

In practice, a qubit is typically a microscopic system, such as an atom, a nuclear spin, or a polarised photon.

As we have already mentioned, any computational step, that is, any physically admissible operation U on a qubit, is described by a (2×2) unitary matrix U . It modifies the state of the qubit as

$$|\psi\rangle \mapsto |\psi'\rangle = U|\psi\rangle$$

which we can write explicitly as

$$\begin{bmatrix} \alpha'_0 \\ \alpha'_1 \end{bmatrix} = \begin{bmatrix} U_{00} & U_{01} \\ U_{10} & U_{11} \end{bmatrix} \begin{bmatrix} \alpha_0 \\ \alpha_1 \end{bmatrix}$$

That is, the operation U turns the state $|\psi\rangle$, with components α_k , into the state $|\psi'\rangle = U|\psi\rangle$, with components $\alpha'_l = \sum_k U_{lk}\alpha_k$.

2.3 Quantum gates and circuits

Atoms, trapped ions, molecules, nuclear spins and many other quantum objects, which we call qubits, can be used to implement simple quantum interference (something which we have still yet to explain), and hence simple quantum computation. There is no need to learn about physics behind these diverse technologies if all you

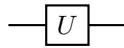
Here we are talking about *isolated* systems. As you will soon learn, a larger class of physically admissible operations is described by completely positive maps. It may sound awfully complicated but, as you will soon see, it is actually very simple.

want is to understand the basics of quantum computation. We may now conveniently forget about any specific experimental realisation of a qubit and just remember that any manipulations on qubits have to be performed by physically admissible operations, and that such operations are represented by unitary transformations.

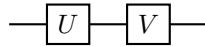
A **quantum (logic) gate** is a device which performs a fixed unitary operation on selected qubits in a fixed period of time, and a **quantum circuit** is a device consisting of quantum logic gates whose computational steps are synchronised in time.

The **size** of such a circuit is the number of gates it contains. The gates in a circuit can be divided into layers, where the gates in the same layer operate at the same time, and the number of such layers is called the **depth** of a circuit.

Some unitary U acting on a single qubit is represented diagrammatically as



This diagram should be read *from left to right*. The horizontal line represents a qubit that is inertly carried from one quantum operation to another. We often call this line a **quantum wire**. The wire may describe translation in space (e.g. atoms travelling through cavities) or translation in time (e.g. a sequence of operations performed on a trapped ion). A sequence of two gates acting on the same qubit, say U followed by V , is represented by



and is described by the matrix product VU (note the order in which we multiply the matrices).

2.4 Single qubit interference

Let us now describe what is probably the most important sequence of operations performed on a single qubit: a generic **single-qubit interference**. It is typically constructed as a sequence of three elementary operations:

1. the **Hadamard gate**
2. a **phase-shift gate**
3. the **Hadamard gate** again.

We represent it graphically as



We sometimes write the phase gate as P_φ instead, if it makes the circuit easier to read.

where the definitions of the Hadamard and phase-shift gates are as in Section 1.6:

Hadamard: $H = \frac{1}{\sqrt{2}} \begin{bmatrix} 1 & 1 \\ 1 & -1 \end{bmatrix}$ $|0\rangle \mapsto \frac{1}{\sqrt{2}}(|0\rangle + |1\rangle)$
 $|1\rangle \mapsto \frac{1}{\sqrt{2}}(|0\rangle - |1\rangle)$

Phase-shift: $P_\varphi = \begin{bmatrix} 1 & 0 \\ 0 & e^{i\varphi} \end{bmatrix}$ $|0\rangle \mapsto |0\rangle$
 $|1\rangle \mapsto e^{i\varphi}|1\rangle$

Note that we sometimes use the notation $|+\rangle$ and $|-\rangle$ when talking about Hadamard gates, where

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

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

You will see this specific sequence of gates over and over again, for it is quantum interference that gives quantum computation additional capabilities.

Something that many explanations of quantum computing say is the following: “quantum computers are quicker because they evaluate all possible solutions at once, in parallel”. **This is not accurate.**

Firstly, quantum computers are not necessarily “quicker” than classical computers, but can simply implement quantum algorithms, *some* of which *are* quicker than their classical counterparts. Secondly, the idea that they “just do all the possible computations at once” is false — instead, they rely on thoughtfully using interference (which can be constructive or destructive) to *modify the probabilities of specific outcomes*.

The motto to keep in mind is that *the power of quantum computing comes from quantum interference*.

Indeed, you have already seen this sequence: recall our study of Ramsey interferometry (Section 1.5), and note how this is essentially the same!

The product of the three matrices $HP_\varphi H$ describes the action of the whole circuit: it gives the transition amplitudes between states $|0\rangle$ and $|1\rangle$ at the input and the output as

$$\frac{1}{\sqrt{2}} \begin{bmatrix} 1 & 1 \\ 1 & -1 \end{bmatrix} \begin{bmatrix} 1 & 0 \\ 0 & e^{i\varphi} \end{bmatrix} \frac{1}{\sqrt{2}} \begin{bmatrix} 1 & 1 \\ 1 & -1 \end{bmatrix} = e^{i\frac{\varphi}{2}} \begin{bmatrix} \cos \varphi/2 & -i \sin \varphi/2 \\ -i \sin \varphi/2 & \cos \varphi/2 \end{bmatrix}$$

Given that our input state is almost always $|0\rangle$, it is sometimes much easier and more instructive to step through the execution of this circuit and follow the evolving state. The interference circuit effects the following sequence of transformations:

$$\begin{aligned} |0\rangle &\xrightarrow{H} \frac{1}{\sqrt{2}}(|0\rangle + |1\rangle) \\ &\xrightarrow{P_\phi} \frac{1}{\sqrt{2}}(|0\rangle + e^{i\phi}|1\rangle) \\ &\xrightarrow{H} \cos \frac{\phi}{2}|0\rangle - i \sin \frac{\phi}{2}|1\rangle. \end{aligned}$$

We ignore the global phase factor $e^{i\frac{\varphi}{2}}$.

The first Hadamard gate prepares an equally weighted superposition of $|0\rangle$ and $|1\rangle$ and the second Hadamard closes the interference by bringing the interfering paths together. The phase shift φ in between effectively controls the entire evolution and determines the output. The probabilities of finding the qubit in state $|0\rangle$ or $|1\rangle$ at the output are, respectively,

$$\Pr(0) = \cos^2 \frac{\phi}{2}$$

$$\Pr(1) = \sin^2 \frac{\phi}{2}.$$

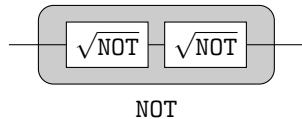
This simple quantum process contains, in a nutshell, the essential ingredients of quantum computation. This sequence (Hadamard–phase shift–Hadamard) will appear over and over again. It reflects a natural progression of quantum computation: first we prepare different computational paths, then we evaluate a function which effectively introduces phase shifts into different computational paths, then we bring the computational paths together at the output.

2.5 The square root of NOT

Now that we have poked our heads into the quantum world, let us see how quantum interference challenges conventional logic. Consider the following task:

Design a logic gate that operates on a single bit and such that when it is followed by another, identical, logic gate the output is always the negation of the input.

Let us call the resulting logic gate **the square root of NOT**, or $\sqrt{\text{NOT}}$.



A simple check, such as an attempt to construct a truth table, should persuade you that there is no such operation in logic. It may seem reasonable to argue that since there is no such operation in logic, $\sqrt{\text{NOT}}$ is impossible. But it does exist! Experimental physicists routinely construct such “impossible” gates in their laboratories. It is a physically admissible operation described by the unitary matrix

$$\sqrt{\text{NOT}} = \frac{1}{2} \begin{bmatrix} 1+i & 1-i \\ 1-i & 1+i \end{bmatrix} = \frac{1}{\sqrt{2}} \begin{bmatrix} e^{i\frac{\pi}{4}} & e^{-i\frac{\pi}{4}} \\ e^{-i\frac{\pi}{4}} & e^{i\frac{\pi}{4}} \end{bmatrix}.$$

There are infinitely many unitary operations that act as the square root of NOT.

Indeed,

$$\frac{1}{2} \begin{bmatrix} 1+i & 1-i \\ 1-i & 1+i \end{bmatrix} \frac{1}{2} \begin{bmatrix} 1+i & 1-i \\ 1-i & 1+i \end{bmatrix} = \begin{bmatrix} 0 & 1 \\ 1 & 0 \end{bmatrix}.$$

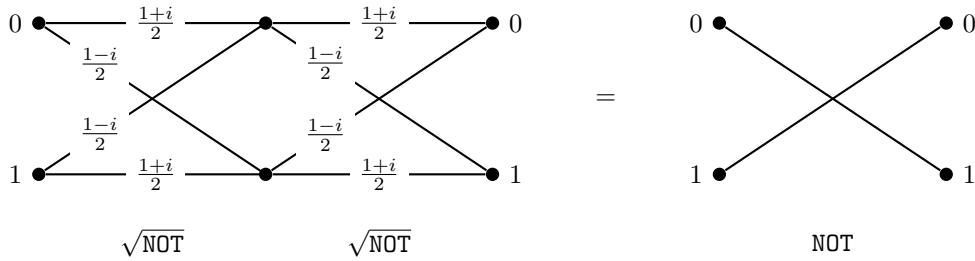


Figure 2.2: A computation that, when repeated, gives exactly NOT. An unlabelled line means that it has probability 1, and the lack of a line corresponds to having probability 0.

We could also step through the circuit diagram and follow the evolution of the state vector:

$$|0\rangle \xrightarrow{\sqrt{\text{NOT}}} \frac{1}{\sqrt{2}} [e^{i\frac{\pi}{4}}|0\rangle + e^{-i\frac{\pi}{4}}|1\rangle] \xrightarrow{\sqrt{\text{NOT}}} |1\rangle$$

Or, if you prefer to work with column vectors and matrices, you can write the two consecutive application of $\sqrt{\text{NOT}}$ to state $|0\rangle$ as

$$\begin{bmatrix} 0 \\ 1 \end{bmatrix} \xleftarrow{\frac{1}{\sqrt{2}} \begin{bmatrix} e^{i\frac{\pi}{4}} \\ e^{-i\frac{\pi}{4}} \end{bmatrix}} \begin{bmatrix} 1 \\ 0 \end{bmatrix}$$

Just remember that circuits diagrams are read from *left to right*, and vector and matrix operations go from *right to left*.

where each “ $\longleftarrow|$ ” denotes multiplication by $\frac{1}{\sqrt{2}} \begin{bmatrix} e^{i\frac{\pi}{4}} & e^{-i\frac{\pi}{4}} \\ e^{-i\frac{\pi}{4}} & e^{i\frac{\pi}{4}} \end{bmatrix}$.

One way or another, quantum theory explains the behaviour of $\sqrt{\text{NOT}}$, and so, reassured by the physical experiments that corroborate this theory, logicians are now entitled to propose a new logical operation $\sqrt{\text{NOT}}$. Why? Because a faithful physical model for it exists in nature!

One such experiment (which we will soon discuss, in Section 3.1) is the so-called Mach-Zehnder interferometer.

2.6 Phase gates galore

We have already met the generic phase gate $P_\varphi = \begin{bmatrix} 1 & 0 \\ 0 & e^{i\varphi} \end{bmatrix}$ which acts via

$$\begin{aligned} |0\rangle &\longmapsto |0\rangle \\ |1\rangle &\longmapsto e^{i\varphi}|1\rangle \end{aligned}$$

but there are three specific examples of P_φ that are important enough to merit their own names (two of which are rather confusing, at first glance).

Phase-flip: $Z = \begin{bmatrix} 1 & 0 \\ 0 & -1 \end{bmatrix}$ $|0\rangle \longmapsto |0\rangle$

$\frac{\pi}{4}$ -phase: $S = \begin{bmatrix} 1 & 0 \\ 0 & i \end{bmatrix}$ $|0\rangle \longmapsto |0\rangle$

$\frac{\pi}{8}$ -phase: $T = \begin{bmatrix} 1 & 0 \\ 0 & e^{i\frac{\pi}{4}} \end{bmatrix}$ $|0\rangle \longmapsto |0\rangle$

$|1\rangle \longmapsto e^{i\frac{\pi}{4}}|1\rangle$

Recall that a phase gate P_φ is only defined up to a global phase factor, and so we can write its matrix either as

$$P_\varphi = \begin{bmatrix} 1 & 0 \\ 0 & e^{i\varphi} \end{bmatrix}$$

or as

$$P_\varphi = \begin{bmatrix} e^{-i\frac{\varphi}{2}} & 0 \\ 0 & e^{i\frac{\varphi}{2}} \end{bmatrix}$$

The first version is more common in the quantum information science community, but the second one is sometimes more convenient to use, as it has determinant 1, and hence belongs to a group called SU(2). We will occasionally switch to the SU(2) version of phase gates, and this is where the $\frac{\pi}{4}$ -phase and $\frac{\pi}{8}$ -phase gates get their names, since their SU(2) versions have $e^{\mp i\pi/4}$ and $e^{\mp i\pi/8}$ (respectively) on the diagonal, even though they are actually of phase $\pi/2$ and $\pi/4$ (respectively).

The groups SU(2) and SO(3).

We will soon explain what this group SU(2) is and how it relates to another important group called SO(3), but it turns up in many places throughout quantum physics, as well as mathematics in general. Other places you might see SU(2) appear are when talking about **quaternions** (which are somehow the next thing in the sequence $\mathbb{R} \hookrightarrow \mathbb{C} \hookrightarrow \mathbb{H}$) and two of the four “fundamental interactions”, namely electromagnetism and the weak nuclear force, which get bundled together into something known as **electroweak interaction**.

We will also eventually talk about how this aforementioned relationship between $SU(2)$ and $SO(3)$ lets us describe *rotations* of things in three-dimensional space. The abstract mathematical concept lying behind this is one with a very lofty-sounding title indeed: **representation theory of Lie algebras**. This lets us formally talk about things like (non-relativistic) **spin**. As for this application of $SU(2)$ in studying the electroweak interaction, this is an example of something known as **gauge theory**.

The remaining gate, the phase-flip Z , is arguably the most important specific phase gate, since it is one of the **Pauli operators**, which we will now discuss.

While we're talking about phase, we should also justify why we keep on saying "let us ignore the global phase factors". In general, states differing only by a global phase are physically indistinguishable, and so it is *physical experimentation* that leads us to this mathematical choice of only defining things up to a global phase.

Global phase.

If you are more mathematically minded, then we can justify ignoring the global phase in a few other ways. Taking the axiomatic approach, where values of physical observables correspond to eigenvalues of operators, think about how the eigenvalues of a matrix A relate to those of the matrix μA , where μ is a complex number with $|\mu| = 1$. One "high-level" way of dealing with this, in the language of gauge theory, is to talk of **invariance under gauge symmetry** (here, in particular, we're talking about **$U(1)$** symmetries).

2.7 Pauli operators

Adding to our collection of common single-qubit gates, we now look at the three **Pauli operators** σ_x , σ_y , and σ_z , also denoted by X , Y , and Z , respectively. These three operators, combined with the identity, satisfy a lot of nice formal properties, which we shall examine briefly here, and then return to in more detail later on, in Section 3.3. After that, these operators will turn up everywhere, so it's good to get familiar with them!

Most of the time we refer to "operators" as "matrices", where the implicit assumption is that we are using the standard basis $\{|0\rangle, |1\rangle\}$.

Identity: $\mathbf{1} = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix} \quad |0\rangle \mapsto |0\rangle \quad |1\rangle \mapsto |1\rangle$

Bit-flip: $X = \begin{bmatrix} 0 & 1 \\ 1 & 0 \end{bmatrix} \quad |0\rangle \mapsto |1\rangle \quad |1\rangle \mapsto |0\rangle$

Bit-phase-flip: $Y = \begin{bmatrix} 0 & -i \\ i & 0 \end{bmatrix} \quad |0\rangle \mapsto i|1\rangle \quad |1\rangle \mapsto -i|0\rangle$

Phase-flip: $Z = \begin{bmatrix} 1 & 0 \\ 0 & -1 \end{bmatrix} \quad |0\rangle \mapsto |0\rangle \quad |1\rangle \mapsto -|1\rangle$

The identity is just a quantum wire, and we have already seen (Section 2.6) the X and Z gates as the bit-flip and phase-flip, respectively. Note that, of the X and Z gates, only the X gate has a classical analogue (namely the logical NOT operator). The remaining gate, the Y operator, describes the combined effect of both the bit- and the phase-flip: $ZX = iY$.

In fact, this is just one of the equations that the Pauli matrices satisfy. The Pauli matrices are unitary and Hermitian, they square to the identity, and they anticommute. By this last point, we mean that

$$XY = -YX,$$

$$XZ = -ZX,$$

$$YZ = -ZY.$$

As already mentioned, they satisfy $ZX = iY$, but also any cyclic permutation of this equation (that is, replace X with Y , Y with Z , and Z with X , and repeat this as many times as you wish).

These operators are also called **sigma operators** (usually when we use the notation σ_x , σ_y , σ_z) or (when written as matrices in the standard basis, as we have done) as **Pauli spin matrices**. They are so ubiquitous in quantum physics that they should certainly be memorised.

2.8 From bit-flips to phase-flips, and back again

The Pauli Z gate is a special case of a phase gate P_φ with $\varphi = \pi$. When we insert it into the interference circuit we obtain

$$\text{---} [H] --- [Z] --- [H] \text{---} = \text{---} [X] \text{---}$$

If you wish to verify this, write the Hadamard gate as $H = (X + Z)/\sqrt{2}$ and use the properties of the Pauli operators. So the Hadamard gate turns phase-flips into bit-flips, but it also turns bit-flips into phase-flips:

$$\text{---} [H] --- [X] --- [H] \text{---} = \text{---} [Z] \text{---}$$

Let us also add, for completeness, that $HYH = -Y$. You will see these identities again and again, especially when we discuss quantum error corrections.

$$HXH = Z$$

$$HZH = X$$

$$HYH = -Y$$

Unitaries, such as H , that take the three Pauli operators to the Pauli operators via conjugation form the so-called **Clifford group**, which we will meet later on, in Chapter 7. Which phase gate is in the Clifford group of a single qubit?

Any complex number z is uniquely specified by two real parameters, writing $z = x + iy$ or $z = re^{i\varphi}$, for example. This is an instance of the fact that \mathbb{C} is a two-dimensional vector space over \mathbb{R} .

Parameter counting.

This sort of argument — counting how many parameters determine a family of matrices — is really an example of calculating the dimension of a vector space. More generally, saying things like “imposing a polynomial equation condition on the coefficients lowers the number of (complex) parameters necessary by 1” is the bread and butter of [algebraic geometry](#), where we try to understand how satisfying polynomial equations can be interpreted as geometrically modifying

high-dimensional “shapes”.

In particular, we need *four* real parameters to specify a (2×2) unitary matrix. If we are prepared to ignore global phase factors (which we are!) then there are only three real parameters left. The real question is, can we use this to construct and implement *any* unitary on a single qubit in some simple way?

Delightfully, the answer is *yes, we can*.

Any unitary operation on a qubit (up to an overall multiplicative phase factor) can be implemented by a circuit containing just two Hadamards and three phase gates, with adjustable phase settings, as in Figure 2.3.



Figure 2.3: The universal circuit for unitary (2×2) matrices, exhibiting how any such matrix is uniquely determined (up to a global phase) by three real parameters.

If we multiply the matrices corresponding to each gate in the network we obtain the single matrix

$$U(\alpha, \beta, \varphi) = \begin{bmatrix} e^{-i(\frac{\alpha+\beta}{2})} \cos \varphi/2 & -ie^{i(\frac{\alpha-\beta}{2})} \sin \varphi/2 \\ -ie^{-i(\frac{\alpha-\beta}{2})} \sin \varphi/2 & e^{i(\frac{\alpha+\beta}{2})} \cos \varphi/2 \end{bmatrix}.$$

Any (2×2) unitary matrix (up to global phase) can be expressed in this form using the three independent real parameters, α , β , and φ , which take values in $[0, 2\pi]$. In order to see that this construction does what it claims, let us explore an intriguing mathematical connection between single-qubit unitaries and rotations in three dimensions.

2.10 The Bloch sphere

Unitary operations on a single qubit form a group. More precisely, the set of all (2×2) unitary matrices forms a (*non-abelian*) group under matrix multiplication, denoted by $U(2)$. It turns out that compositions of single-qubit unitaries behave pretty much the same as compositions of rotations in three dimensions. Technically speaking, we claim that $U(2)/U(1) \cong SO(3)$. That is, (2×2) unitaries, up to global phase, form a group which is isomorphic to the group of rotations in three dimensions, which is denoted by $SO(3)$. This isomorphism helps to visualise the actions of single-qubit gates.

There are many ways to introduce this isomorphism. Here we will just show how to represent single-qubit state vectors in terms of Euclidean vectors in three dimensions; later (in Section 3.4) we will actually relate unitary operations on state vectors to rotations in this Euclidean space, demonstrating this isomorphism.

Any single-qubit state can be written as $|\psi\rangle = \alpha|0\rangle + \beta|1\rangle$, constrained by the relation $|\alpha|^2 + |\beta|^2 = 1$. This suggests a more natural parametrisation as

$$|\psi\rangle = \cos(\theta/2)e^{i\varphi_0}|0\rangle + \sin(\theta/2)e^{i\varphi_1}|1\rangle$$

(note that there is a good reason to use $\theta/2$ instead of θ , and we will explain why later on). We can then factor out a global phase:

$$|\psi\rangle = e^{i\varphi_0} (\cos(\theta/2)|0\rangle + \sin(\theta/2)e^{i\varphi_1}|1\rangle),$$

and even remove it completely, since states that are identical up to a global phase are physically indistinguishable.

The parametrisation in terms of θ and φ should remind you (if you are familiar with it) of [spherical polar coordinates](#) for the surface of a sphere.

Remember that the order of matrix multiplication is reversed when compared to reading circuit diagrams.

Note that $U(1) \cong \mathbb{C}^\times$, where \mathbb{C}^\times is the multiplicative group of invertible elements of the complex numbers, i.e. the set $\mathbb{C} \setminus \{0\}$ with the group operation given by multiplication.

That is, we have the group $U(2)$ acting on the space of single-qubit state vectors, and we have the group $SO(3)$ acting on the unit sphere $S^2 \subset \mathbb{R}^3$. In this chapter we will discuss how to go from one *space* (i.e. the thing being acted upon) to the other; in Section 3.4 we will discuss how to go from one *group* (i.e. the thing doing the acting) to the other.

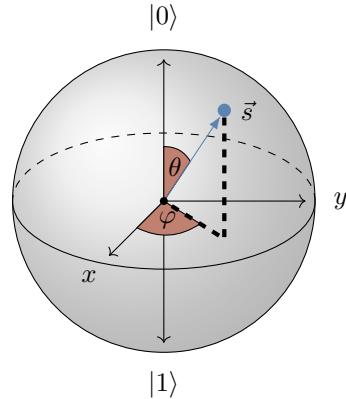


Figure 2.4: The Bloch sphere, with the point \vec{s} corresponding to $|\psi\rangle$ marked.

We call this sphere the **Bloch sphere**, and the unit vector \vec{s} defined by θ and φ the **Bloch vector**. This is a very useful way to visualise quantum states of a single qubit and unitary operations that we perform on it. Any unitary action on the state vector will induce a rotation of the corresponding Bloch vector. But what kind of rotation?

We give a complete answer to this question soon, in Section 3.4, but we might as well give some specific results here first, since some are easy enough to calculate “by hand”. Here is one fundamental observation: *any two orthogonal state vectors appear on the Bloch sphere as two Bloch vectors pointing in opposite directions*. Now, the two eigenvectors of a single-qubit unitary U are always orthogonal, and so must define an axis running through the centre of the Bloch sphere. *This is the axis about which the Bloch vector is rotated when U acts on the corresponding state vector.* The rotation angle α is given by the eigenvalues of U , which, up to a global phase factor, are of the form $e^{\mp i\alpha/2}$.

It is instructive to work out few simple cases and get a feel for the rotations corresponding to the most common unitaries. For example, it is easy to check that a phase gate P_α acts by

$$\cos \frac{\theta}{2} |0\rangle + e^{i\varphi} \sin \frac{\theta}{2} |1\rangle \mapsto \cos \frac{\theta}{2} |0\rangle + e^{i(\varphi+\alpha)} \sin \frac{\theta}{2} |1\rangle.$$

The azimuthal angle changes from φ to $\varphi + \alpha$, and so the Bloch sphere is rotated anticlockwise by α about the z -axis. The Bloch vectors corresponding to the two eigenvectors of P_α , namely $|0\rangle$ and $|1\rangle$, define the axis of the rotation.

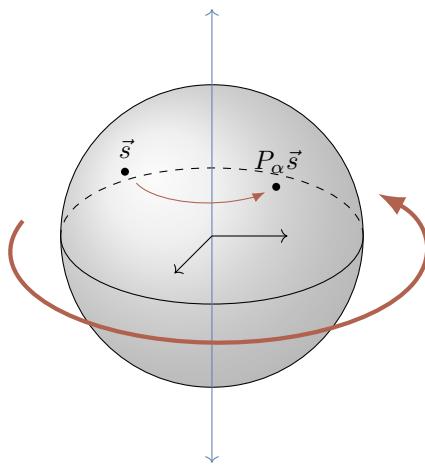


Figure 2.5: Phase gates P_α represent rotations of the Bloch sphere around the z -axis.

As previously mentioned, the Pauli operator $Z = \sigma_z$ is a special case of a phase gate, and represents rotation by 180° (that is, π radians), about the z -axis. You can also verify that $X = \sigma_x$, with eigenvectors $(|0\rangle \pm |1\rangle)/\sqrt{2}$, represents rotation by 180° about the x -axis, and $Y = \sigma_y$, with eigenvectors $(|0\rangle \pm i|1\rangle)/\sqrt{2}$, represents rotation by 180° about the y -axis. Again, note that, by the definition of the axis, the points of intersection of these axes with the Bloch sphere are exactly the eigenvectors of the operator.

How about the Hadamard gate? Like the Pauli operators, it squares to the identity ($H^2 = 1$), which implies that its eigenvalues are ± 1 . Thus it will correspond to a rotation by 180° . But about which axis? This time, rather than finding eigenvectors of H , we notice that $H X H = Z$ and $H Z H = X$, thus H must swap the x - and z -axes, turning rotations about the z -axis into rotations about the x -axis, and vice versa. The Hadamard gate must then represent rotation by 180° about the diagonal $(x+z)$ -axis. You may also notice that, after this rotation, the y -axis points in the opposite direction, which seems to be related to another identity: $HYH = -Y$. This is not a coincidence!

We will eventually show that the effect of the rotation represented by unitary U on the Bloch vector with components s_x, s_y, s_z is summarised in the formula

$$U(s_x X + s_y Y + s_z Z)U^\dagger = s'_x X + s'_y Y + s'_z Z,$$

where s'_x, s'_y , and s'_z are the components of the rotated Bloch vector.

2.11 Drawing points on the Bloch sphere

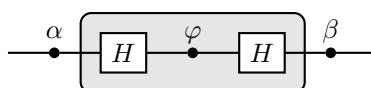
We know that the state $|0\rangle$ corresponds to the north pole of the Bloch sphere, and the state $|1\rangle$ to the south, but what about an arbitrary state $|\psi\rangle = \alpha|0\rangle + \beta|1\rangle$? By definition, we can find the parametrisation in terms of θ and φ , but there is also a neat “trick” for finding the point on the Bloch sphere that corresponds to $|\psi\rangle$, which goes as follows.

1. Calculate $\lambda = \beta/\alpha$ (assuming that $\alpha \neq 0$, since otherwise $|\psi\rangle = |1\rangle$).
2. Write $\lambda = \lambda_x + i\lambda_y$ and mark the point $p = (\lambda_x, \lambda_y)$ in the xy -plane (i.e. the plane $\{z = 0\}$).
3. Draw the line going through the south-pole (which corresponds to $|1\rangle$) and the point p . This will intersect the Bloch sphere in exactly one other point, and this is exactly the point corresponding to $|\psi\rangle$.

Note that this lets you *draw* the point on the sphere, but doesn’t (immediately) give you the *coordinates* for it. That is, this method is nice for geometric visualisation, but the parametrisation method is much better when it comes to actually doing calculations.

2.12 Composition of rotations

We are now in a position to understand the circuit in Figure 2.3 in geometric terms. It is a [very useful fact of geometry](#) (which we shall take for granted) that *any* rotation in three-dimensional Euclidean space can be performed as a sequence of three specific rotations: one about the z -axis, one about the x -axis, and one more about z -axis. The circuit does exactly this:



The first phase gate effects rotation by α about the z -axis, the second phase gate is sandwiched between the two Hadamard gates, and these three gates together effect rotation by φ about the x -axis, and, finally, the third phase gate effects rotation by β about the z -axis. So we can implement any unitary U by choosing the three phase shifts, α, φ , and β , which are known as the three [Euler angles](#).

2.13 A finite set of universal gates

The Hadamard gate and the phase gates, with adjustable phases, allow us to implement an arbitrary single-qubit unitary *exactly*. The tacit assumption here is that we have *infinitely many* phase gates: one gate for each phase. In fact, we can pick just one phase gate, namely any phase gate P_α with the phase α that is incommensurate with π . It is clear that repeated iteration of P_α can be used to approximate any other phase gate to arbitrary accuracy: indeed, rotate the Bloch sphere by α about the z -axis sufficiently many times and you end up as close as you please to any other rotation about the z -axis.

If you want to be ε -close to the desired angle of rotation, then you may need to repeat the rotation by α roughly $1/\varepsilon$ times. Indeed, within n applications (for $n\alpha \gg 2\pi$) of P_α , we expect the accessible angles to be approximately evenly distributed within the range $[0, 2\pi]$, i.e. any angle of rotation can be achieved to an accuracy of $\varepsilon = 2\pi/n$ by using up to $n \approx 1/\varepsilon$ applications of P_α . So we can use *just one* phase gate to approximate the *three* phase gates in the circuit in Figure 2.3.

There are other ways of implementing irrational rotations of the Bloch sphere. For example, take the Hadamard gate and the T gate (also known as the $\pi/8$ -phase gate, despite being the phase gate P_φ for $\varphi = \pi/4$ as we saw earlier in Section 2.6). You can check that the compositions $THTH$ and $HTHT$ represent rotations by angles which are irrational multiples of π , about two different axes. We can then compose a sequence of these two rotations to approximate any other rotation of the sphere. This may look very nice in theory, but there are issues with the actual physical implementation of this approach: in reality, all the gates in the circuit will operate with only *finite* precision, and the phase gates will deviate from implementing the required *irrational* rotations. It turns out, however, that we can tolerate minor imperfections; the final result will not be that far off.

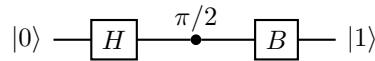
2.14 Remarks and exercises

2.14.1 One simple circuit

Let B be the matrix

$$B = \frac{1}{\sqrt{2}} \begin{bmatrix} 1 & i \\ i & 1 \end{bmatrix}.$$

1. Show that B is unitary.
2. Find the overall unitary corresponding to the circuit



If you get the answer $\begin{bmatrix} 0 & 1 \\ i & 0 \end{bmatrix}$ then you performed the matrix multiplication in the wrong order!

2.14.2 Change of basis

Write the state

$$|\psi\rangle = \frac{3}{5}|0\rangle + \frac{4}{5}|1\rangle$$

as a superposition of the states $|\pm\rangle = (|0\rangle \pm |1\rangle)/\sqrt{2}$. In other words, find $\alpha, \beta \in \mathbb{C}$ such that

$$|\psi\rangle = \alpha|+\rangle + \beta|-\rangle.$$

2.14.3 Operators as unitary matrices

Say we have a linear operator that acts on a qubit as follows:

$$\begin{aligned} |0\rangle &\mapsto \alpha|0\rangle + \beta|1\rangle \\ |1\rangle &\mapsto \beta^*|0\rangle - \alpha^*|1\rangle. \end{aligned}$$

1. Show that this linear operator is equal to

$$M = (\alpha|0\rangle + \beta|1\rangle)(\langle 0| + (\beta^*|0\rangle - \alpha^*|1\rangle)\langle 1|)$$

by calculating $M|0\rangle$ and $M|1\rangle$.

2. If we pick the standard basis

$$|0\rangle \equiv \begin{bmatrix} 1 \\ 0 \end{bmatrix} \quad |1\rangle \equiv \begin{bmatrix} 0 \\ 1 \end{bmatrix}$$

then what is the matrix representation of M ?

3. Verify that M is unitary.

2.14.4 Completing an orthonormal basis

Imagine that some quantum system has four energy levels: $|0\rangle$, $|1\rangle$, $|2\rangle$, and $|3\rangle$. Consider the three orthonormal vectors

$$|\psi_1\rangle = \frac{1}{\sqrt{3}}(|0\rangle + |1\rangle + |2\rangle)$$

$$|\psi_2\rangle = \frac{1}{\sqrt{2}}(|0\rangle - |2\rangle)$$

$$|\psi_3\rangle = |3\rangle.$$

Find a vector $|\psi_4\rangle$ such that $\{|\psi_1\rangle, |\psi_2\rangle, |\psi_3\rangle, |\psi_4\rangle\}$ is an orthonormal basis.

2.14.5 Some sums of inner products

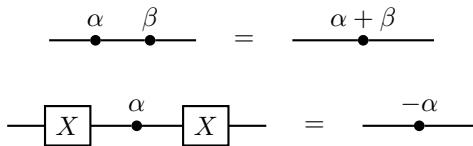
Let $\{|v_1\rangle, \dots, |v_N\rangle\}$ be an orthonormal basis. Evaluate the following:

1. $\sum_{k=1}^{N-1} k^2 \langle v_{N-1} | v_k \rangle$
2. $\sum_{k=1}^{N-1} k^2 \langle v_N | v_k \rangle$
3. $\sum_{k=1}^N \langle v_{N-1} | v_k \rangle \langle v_k | v_{N-1} \rangle$
4. $\sum_{j,k=1}^{N-1} \langle v_j | v_k \rangle$

All of your answers should be numbers (in fact, they'll all even be integers).

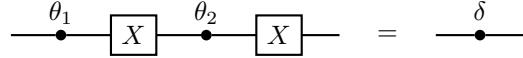
2.14.6 Some circuit identities

1. Prove the following circuit identities, ignoring any global phase:



2. Using the above, find θ_1 and θ_2 such that

$$\text{---} \bullet^{\theta_1} \bullet^{\theta_2} \text{---} = \text{_____}$$



for some value δ .

3. We can think of the second identity in the previous question as an implementation of “if the two X gates are absent then do nothing; otherwise implement the phase gate P_δ ”. Given that unitary matrices are normal, adapt this circuit so that implements “if the two X gates are absent then do nothing; otherwise implement the unitary U ” where U is a one-qubit unitary.

2.14.7 Unitaries preserve length

Let M be a linear operator that sends an orthonormal basis $\{|u_1\rangle, \dots, |u_N\rangle\}$ to a set of states $\{|v_1\rangle, \dots, |v_n\rangle\}$ where each $|v_i\rangle$ is of length 1.

1. Show that the length of the vector $M(|u_1\rangle + |u_2\rangle)/\sqrt{2}$ is

$$\sqrt{1 + \text{Re}\langle v_1 | v_2 \rangle}.$$

2. Find a correctly normalised superposition $|\psi\rangle = \lambda(\alpha|u_1\rangle + \beta|u_2\rangle)$ such that $M|\psi\rangle$ is of length 1

$$\sqrt{1 + \text{Im}\langle v_1 | v_2 \rangle}.$$

3. Using the above, show that, for all $|\psi\rangle$ of length 1, the vector $M|\psi\rangle$ is of length 1 if and only if M is unitary.

2.14.8 Unknown phase

Consider the usual quantum interference circuit:



Suppose you can control the input of the circuit and measure the output, but you do not know the phase shift φ introduced by the phase gate. You prepare input $|0\rangle$ and register output $|1\rangle$. What can you say about φ ?

Now you are promised that φ is either 0 or π . You can run the circuit *only once* to find out which of the two phases was chosen. Is it possible to then always correctly guess whether φ was 0 or π ?

This problem forms the basis for a lot of material later on: most quantum algorithms build upon it. We will return to it again and again in Chapter 10.

2.14.9 One of the many cross-product identities

When working with three-dimensional geometry, the [cross product](#) of vectors is very useful, so here is an exercise to help you get used to working with it.

Let's say we want to derive the identity

$$(\vec{a} \cdot \vec{\sigma})(\vec{b} \cdot \vec{\sigma}) = (\vec{a} \cdot \vec{b})\mathbf{1} + i(\vec{a} \times \vec{b}) \cdot \vec{\sigma}.$$

First, notice that the products of Pauli matrices can be written succinctly as

$$\sigma_i \sigma_j = \delta_{ij} \mathbf{1} + i \varepsilon_{ijk} \sigma_k,$$

Hint: all you need here are the Pauli matrices' commutation and anticommutation relations, but it is instructive to derive the identity using the component notation, and below we give a sketch of how such a derivation would go.

where δ_{ij} is **Kronecker delta** (equal to 0 if $i \neq j$, and to 1 if $i = j$) and ε_{ijk} is the **Levi-Civita symbol**:

$$\varepsilon_{ijk} = \begin{cases} +1 & \text{if } (i, j, k) \text{ is } (1, 2, 3), (2, 3, 1), \text{ or } (3, 1, 2) \\ -1 & \text{if } (i, j, k) \text{ is } (3, 2, 1), (1, 3, 2), \text{ or } (2, 1, 3) \\ 0 & \text{if } i = j, \text{ or } j = k, \text{ or } k = i \end{cases}$$

That is, ε_{ijk} is 1 if (i, j, k) is an even permutation of $(1, 2, 3)$, it is -1 if it is an odd permutation, and it is 0 if any index is repeated. The Levi-Civita symbol is anti-symmetric, meaning when any two indices are changed, its sign alternates. Then recall that the scalar (dot) product and vector (cross) product of two Euclidean vectors \vec{a} and \vec{b} can be written, in terms of the components, as

$$\vec{a} \cdot \vec{b} = \sum_{i=1}^3 a_i b_i$$

$$(\vec{a} \times \vec{b})_i = \sum_{j,k=1}^3 \varepsilon_{ijk} a_j b_k.$$

The rest is rather straightforward:

$$(\vec{a} \cdot \vec{\sigma})(\vec{b} \cdot \vec{\sigma}) = \sum_{i,j} a_i b_j \sigma_i \sigma_j = \dots$$

3 Quantum gates

About understanding the square root of NOT via a physical implementation using symmetric beam-splitters. More about the Bloch sphere, via the omnipresent Pauli matrices, which can be described in a more algebraic way.

Before introducing too many new ideas, we first want to study two things we've already seen in more depth, namely the square root of NOT, and the Bloch sphere.

The goal for the latter is to be able to visualise sequences of unitary operations on a qubit as sequences of rotations, and to see the action of some quantum circuits without getting engrossed in lengthy calculations; this also leads us back to the question of universal sets of gates. The goal for the former is to study a way of implementing this gate using physical experiments, and then studying a related construction (the so-called **Mach-Zehnder interferometer**).

3.1 Beam-splitters: physics against logic

A **symmetric beam-splitter** is a cube of glass which reflects half the light that impinges upon it, while allowing the remaining half to pass through unaffected. For our purposes it can simply be viewed as a device that has two input and two output ports, which we label with $|0\rangle$ and $|1\rangle$ as in Figure 3.1.

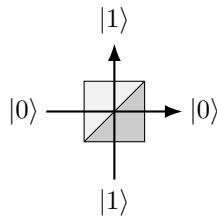


Figure 3.1: A symmetric beam-splitter, with input ports on the bottom and the left sides, and output ports on the top and the right sides.

When we aim a single photon at such a beam-splitter using one of the input ports, we notice that the photon doesn't split in two: we can place photo-detectors wherever we like in the apparatus, fire in a photon, and verify that if any of the photo-detectors registers a hit, none of the others do. In particular, if we place a photo-detector behind the beam-splitter in each of the two possible exit beams, the photon is detected with equal probability at either detector, no matter whether the photon was initially fired into input port $|0\rangle$ or $|1\rangle$.

If we fire the photon into the input port $|0\rangle$, it may seem obvious that, at the very least, the photon is *either* in the **transmitted** beam $|0\rangle$ or in the **reflected** beam $|1\rangle$ during any one run of this experiment. Thus we may be tempted to think of the beam-splitter as a random binary switch which, with equal probability, transforms any binary input into one of the two possible outputs. However, as you might expect (now having already learnt about the double-slit experiment), *this is not necessarily the case*. Let us introduce a second beam-splitter and place two normal mirrors so that both paths intersect at the second beam-splitter, as well as putting a detector at each output port of the second beam-splitter (see Figure 3.2).

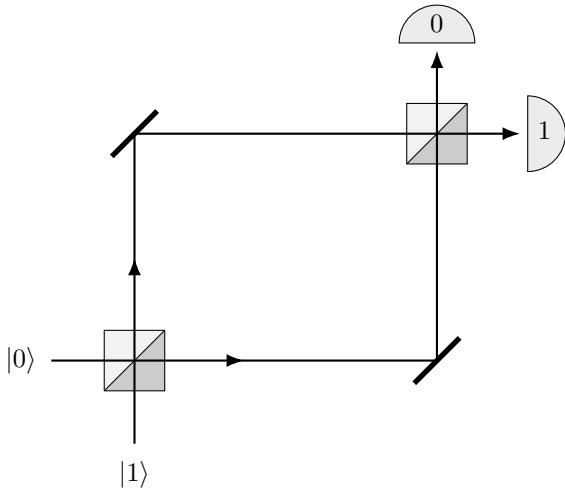


Figure 3.2: Two beam-splitters with mirrors, arranged so that the photon travels through both, along with two detectors. We label the detectors in such a way that, if a photon enters input $|j\rangle$ and is *transmitted* (not reflected) through both beam-splitters, then it is detected by detector j .

Recall the Kolmogorov additivity axiom in classical probability theory: whenever something can happen in several alternative ways, we add probabilities for each way considered separately. We might argue that a photon fired into the input port $|0\rangle$ can reach the detector 0 in two *mutually exclusive* ways: either by two consecutive reflections or by two consecutive transmissions. Each reflection happens with probability $1/2$, and each transmission happens with probability $1/2$, so the total probability of a photon fired into input $|0\rangle$ reaching detector 0 is the sum of the probability of the two consecutive reflections ($1/2 \times 1/2 = 1/4$) and the probability of the two consecutive transmissions ($1/2 \times 1/2 = 1/4$), which gives a probability of $1/2$. This is summarised in Figure 3.3, and makes perfect sense — a random switch followed by a random switch should give nothing else but a random switch.

However, if we set up such an experiment in a lab, *this is not what happens!*

There is no reason why probability theory (or any other *a priori* mathematical construct for that matter) should make any meaningful statements about outcomes of physical experiments.

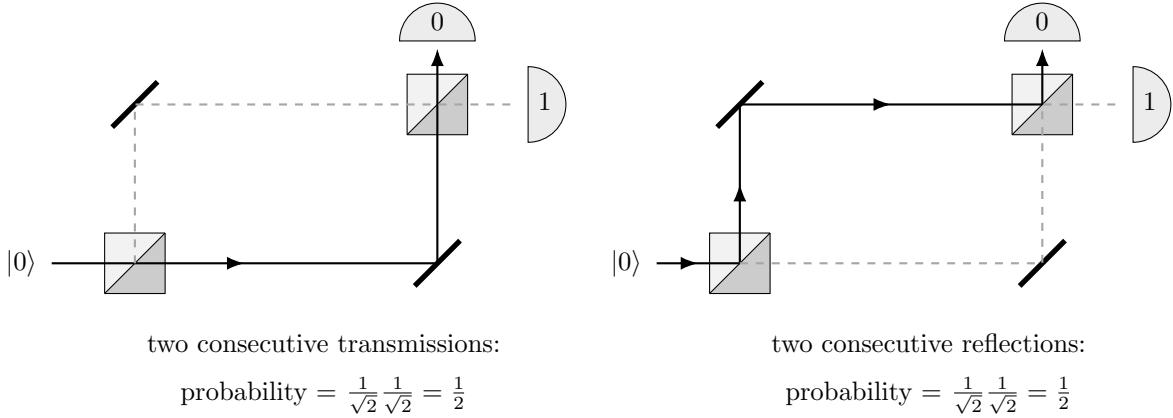


Figure 3.3: The two possible classical scenarios. Note that this is *not* what actually happens in the real physical world!

In experimental reality, when the optical paths between the two beam-splitters are the same, the photon fired from input port $|0\rangle$ *always* strikes detector 1 and *never* detector 0 (and the photon fired from input port $|1\rangle$ *always* strikes detector 0 and *never* detector 1). In other words, *a beam-splitter is a physical implementation of a $\sqrt{\text{NOT}}$ gate.*

The action of the beam-splitter — in fact, the action of any quantum device — can be described by tabulating the amplitudes of transitions between its input and output ports.

$$B = \begin{bmatrix} B_{00} & B_{01} \\ B_{10} & B_{11} \end{bmatrix} = \begin{bmatrix} \frac{1}{\sqrt{2}} & \frac{i}{\sqrt{2}} \\ \frac{i}{\sqrt{2}} & \frac{1}{\sqrt{2}} \end{bmatrix}.$$

The matrix element B_{lk} , where $k, l = 0, 1$, represents the amplitude of transition from input $|k\rangle$ to output $|l\rangle$ (watch the order of indices). Each reflection (entries B_{01} and B_{10}) happens with amplitude $i/\sqrt{2}$, and each transmission (entries B_{00} and B_{11}) happens with amplitude $1/\sqrt{2}$. So the total amplitude that a photon fired from input port $|0\rangle$ will reach detector 0 is the sum of the amplitude of the two consecutive reflections ($i/\sqrt{2} \times i/\sqrt{2} = -1/2$) and the amplitude of the two consecutive transmissions ($1/\sqrt{2} \times 1/\sqrt{2} = 1/2$) which gives the total amplitude 0, and thus a resulting probability of zero.

Note that gate B is not the same square root of NOT as the one we have already seen. In fact, there are infinitely many ways of implementing this “impossible” logical operation.

Unlike probabilities, amplitudes can cancel each other out, witnessing destructive interference.

We can now go on and calculate the amplitude that the photon will reach detector 1. In this case we will get i , which gives probability 1 (since $|i|^2 = 1$). We can then switch to input $|1\rangle$ and repeat our calculations. All possible paths and associated amplitudes are shown in Figure 3.4.

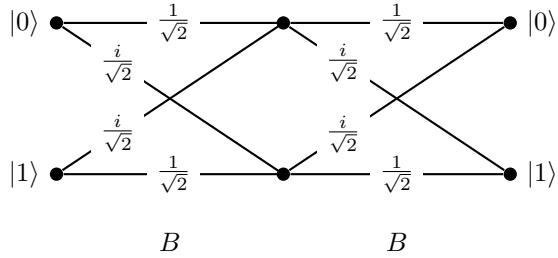


Figure 3.4: All possible transitions and their amplitudes when we compose two beam-splitters, as described by the matrix B above.

However, instead of going through all the paths in this diagram and linking specific inputs to specific outputs, we can simply multiply the transition matrices:

$$BB = \begin{bmatrix} \frac{1}{\sqrt{2}} & \frac{i}{\sqrt{2}} \\ \frac{i}{\sqrt{2}} & \frac{1}{\sqrt{2}} \end{bmatrix} \begin{bmatrix} \frac{1}{\sqrt{2}} & \frac{i}{\sqrt{2}} \\ \frac{i}{\sqrt{2}} & \frac{1}{\sqrt{2}} \end{bmatrix} = \begin{bmatrix} 0 & i \\ i & 0 \end{bmatrix} = iX$$

where

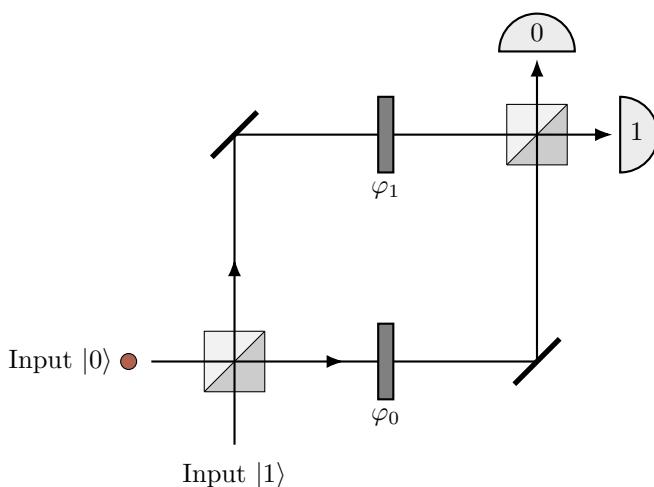
$$X = \text{NOT} = \begin{bmatrix} 0 & 1 \\ 1 & 0 \end{bmatrix}.$$

Bit-flip: $\text{NOT} \equiv X \quad \begin{bmatrix} 0 & 1 \\ 1 & 0 \end{bmatrix}$

Beam-splitter: $\sqrt{\text{NOT}} \equiv B \quad \frac{1}{\sqrt{2}} \begin{bmatrix} 1 & i \\ i & 1 \end{bmatrix}$

3.2 Beam-splitters: quantum interference, revisited

One of the simplest quantum devices in which quantum interference can be controlled is a **Mach–Zehnder interferometer** — see Figure 3.5.



You can play around with a virtual Mach–Zehnder interferometer at [Quantum Flytrap's Virtual Lab](#). (There are also lots of other things you can do in this virtual lab — [go have a look!](#)).

Figure 3.5: The **Mach–Zehnder interferometer**, with the input photon represented by the coloured dot. This experimental set-up is named after the physicists [Ludwig Mach](#) and [Ludwig Zehnder](#), who proposed it back in 1890s.

This is a slightly modified version of the construction shown in Figure 3.2, where we have added two slivers of glass of different thickness into each of the optical paths connecting the two beam-splitters. The slivers are usually referred to as “phase shifters”, and their thicknesses φ_0 and φ_1 are measured in units of the photon’s wavelength multiplied by 2π . These phase shifters are so called because they modify the probability amplitudes by phase factors $e^{i\varphi_0}$ and $e^{i\varphi_1}$, respectively. The only other change that we make is replacing the *symmetric* beam-splitters with **non-symmetric** ones, i.e. they no longer transmit or reflect with equal probability, but instead reflect with some (fixed) probability amplitude $i\sqrt{R}$ and transmit with some probability amplitude \sqrt{T} , where $R + T = 1$. As before, the two input ports of the interferometer are labelled as $|0\rangle$ and $|1\rangle$, and each of the two output ports, also labelled as 0 and 1, terminates in a photodetector.

A photon (the coloured dot in the figure) impinges on the first beam-splitter from one of the two input ports (here input $|0\rangle$) and begins its journey towards one of the two photodetectors. Let U_{ij} denote the probability amplitude that the photon initially in input port $j = 0, 1$ ends up in detector $i = 0, 1$.

In quantum theory we almost always work with probability *amplitudes*, and only once we have a full expression for the amplitude of a given outcome do we square its absolute value to get the corresponding probability.

We will often use i as an index even though it is also used for the imaginary unit. Hopefully, no confusion will arise for it should be clear from the context which one is which.

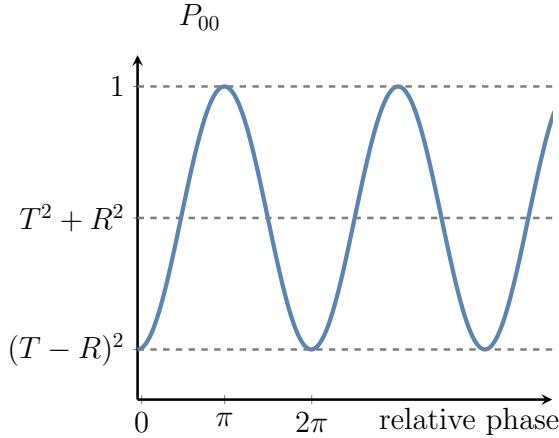
For example, let us calculate U_{00} . We notice that there are two alternative ways for the photon in the input port $|0\rangle$ to end up at the output port 0: It can take the lower path, through the phase shifter φ_0 , or the upper path, through the phase shifter φ_1 . The lower path implies two consecutive transmissions at the beam-splitters and the phase factor $e^{i\varphi_0}$, whereas the upper path implies two consecutive reflections and the phase factor $e^{i\varphi_1}$. Once the photon ends in the output port 0 there is no way of knowing which path was taken, so we add the amplitudes pertaining to each path. The resulting amplitude is

$$U_{00} = \sqrt{T}e^{i\varphi_0}\sqrt{T} + i\sqrt{R}e^{i\varphi_1}i\sqrt{R},$$

and the corresponding probability $P_{00} = |U_{00}|^2$ is

$$\begin{aligned} P_{00} &= \left| \sqrt{T}e^{i\varphi_0}\sqrt{T} + i\sqrt{R}e^{i\varphi_1}i\sqrt{R} \right|^2 \\ &= |Te^{i\varphi_0} - Re^{i\varphi_1}|^2 \\ &= T^2 + R^2 - 2TR \cos(\varphi_1 - \varphi_0). \end{aligned}$$

The “classical” part of this expression, $T^2 + R^2$, basically says that the photon undergoes either two consecutive transmissions with probability T^2 , or two consecutive reflections with probability R^2 . The probability of being transmitted through any phase shifter is always 1, hence the phase shifters play no role in the classical description of this process. But the classical description is *not* correct — it doesn’t agree with physical experiments! — whence the interference term $2TR \cos(\varphi_1 - \varphi_0)$ in which the phase shifters play the essential role. Depending on the *relative* phase $\varphi = \varphi_1 - \varphi_0$ the probability that the detector 0 “clicks” after having fired a photon into input $|0\rangle$ can vary from $(T - R)^2$ to 1.



If we do not care about the experimental details, we can represent the action of the Mach–Zehnder interferometer in terms of a diagram, as in Figure 3.6.

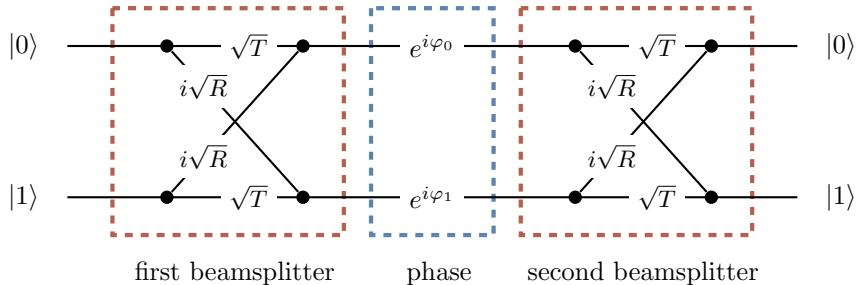


Figure 3.6: The Mach–Zehnder interferometer represented as an abstract diagram.

Here we can follow, from left to right, the multiple different paths that a photon can take in between specific input and output ports. The amplitude along any given path is just the product of the amplitudes pertaining to the path segments (Rule 1, Section 1.1), while the overall amplitude is the sum of the amplitudes for the many different paths (Rule 2, Section 1.1). You can, for example, see that the probability amplitude U_{10} is given by

$$U_{10} = \sqrt{T}e^{i\varphi_0}i\sqrt{R} + i\sqrt{R}e^{i\varphi_1}\sqrt{T}$$

and the corresponding probability is

$$\begin{aligned} P_{10} &= \left| \sqrt{T}e^{i\varphi_0}i\sqrt{R} + i\sqrt{R}e^{i\varphi_1}\sqrt{T} \right|^2 \\ &= 2RT + 2RT \cos(\varphi_1 - \varphi_0). \end{aligned}$$

Again, the first term is of “classical” origin and represents probabilities corresponding to each path: one reflection followed by one transmission plus one transmission followed by one reflection, that is, $RT + TR = 2RT$. The second term is the interference term. Clearly, the photon entering port $|0\rangle$ will end up in one of the two detectors, hence

$$P_{00} + P_{10} = R^2 + 2RT + T^2 = (T + R)^2 = 1.$$

The action of the interferometer is thus fully described by the four probability amplitudes U_{ij} ($i, j = 0, 1$).

The most popular instance of a Mach–Zehnder interferometer involves only *symmetric* beam-splitters (i.e. $R = T = \frac{1}{2}$) and is fully described by the matrix

Any isolated quantum device can fully be described by the matrix of probability amplitudes U_{ij} that input j generates output i .

Really, when you write down the matrices describing the action of the symmetric beam-splitters and the phase gates, and then multiply them all together (which is an exercise worth doing!), you actually obtain $ie^{i\frac{\varphi_0+\varphi_1}{2}}U$ rather than U , but as we have already said, we can ignore global phase factors.

$$U = \begin{bmatrix} -\sin \varphi/2 & \cos \varphi/2 \\ \cos \varphi/2 & \sin \varphi/2 \end{bmatrix}$$

where $\varphi = \varphi_1 - \varphi_0$.

3.3 The Pauli matrices, algebraically

Matrices (of a fixed size, with entries in a fixed field) form a vector space: you can add them, and you can multiply them by a scalar. One possible choice of a basis in the vector space of (2×2) matrices (over any field) is the set of matrices $\{M_{00}, M_{01}, M_{10}, M_{11}\}$, where the entries of M_{ij} are all 0 except for the ij -th entry, which is 1 (e.g. $M_{01} = \begin{bmatrix} 0 & 1 \\ 0 & 0 \end{bmatrix}$). However, it turns out that there is a different basis which offers lots of insights into the structure of the general single-qubit unitary transformations, namely $\{\mathbf{1}, X, Y, Z\}$, i.e. the identity matrix and the three Pauli matrices. We have already defined the Pauli operators (Section 2.7), but we recall their definition here.

Identity:	$\mathbf{1} = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}$	$ 0\rangle \mapsto 0\rangle$	$ 1\rangle \mapsto 1\rangle$
Bit-flip:	$X = \begin{bmatrix} 0 & 1 \\ 1 & 0 \end{bmatrix}$	$ 0\rangle \mapsto 1\rangle$	$ 1\rangle \mapsto 0\rangle$
Bit-phase-flip:	$Y = \begin{bmatrix} 0 & -i \\ i & 0 \end{bmatrix}$	$ 0\rangle \mapsto i 1\rangle$	$ 1\rangle \mapsto -i 0\rangle$
Phase-flip:	$Z = \begin{bmatrix} 1 & 0 \\ 0 & -1 \end{bmatrix}$	$ 0\rangle \mapsto 0\rangle$	$ 1\rangle \mapsto - 1\rangle$

In this chapter we are concerned only with the *single-qubit* Pauli operators. There are analogous *multi-qubit* Pauli operators, but be careful: these do not satisfy all the same properties! For example, anticommutativity (explained below) is special to the *single-qubit* case.

Recall that the Pauli operators (as well as the identity operator) are unitary and Hermitian, square to the identity, and anticommute.

The fact that $\{\mathbf{1}, X, Y, Z\}$ forms a basis for the space of (2×2) complex matrices is equivalent to the statement that *any* (2×2) complex matrix A has a *unique* expansion in the form

$$\begin{aligned} A &= \begin{bmatrix} a_0 + a_z & a_x - ia_y \\ a_x + ia_y & a_0 - a_z \end{bmatrix} \\ &= a_0 \mathbf{1} + a_x \sigma_x + a_y \sigma_y + a_z \sigma_z. \end{aligned}$$

for some complex numbers a_0, a_x, a_y , and a_z .

If we define vectors $\vec{a} = (a_x, a_y, a_z)$ and $\vec{\sigma} = (\sigma_x, \sigma_y, \sigma_z)$, then we can write the above expansion very concisely:

$$A = a_0 \mathbf{1} + \vec{a} \cdot \vec{\sigma}.$$

The algebraic properties of the Pauli matrices can then be neatly compacted (see Exercise 3.7.4) into a single expression:

The multiplication rule:

$$(\vec{a} \cdot \vec{\sigma}) (\vec{b} \cdot \vec{\sigma}) = (\vec{a} \cdot \vec{b}) \mathbf{1} + i(\vec{a} \times \vec{b}) \cdot \vec{\sigma}.$$

Recall that the **trace** of a square matrix A , denoted by $\text{tr } A$, is defined to be the sum of the elements on the main diagonal of A , and defines a linear mapping: for any

Anticommutativity says that

$$\begin{aligned} XY + YX &= 0, \\ XZ + ZX &= 0, \\ YZ + ZY &= 0. \end{aligned}$$

scalars α and β ,

$$\mathrm{tr}(\alpha A + \beta B) = \alpha \mathrm{tr} A + \beta \mathrm{tr} B.$$

Moreover, the trace is invariant under *cyclic* permutations: e.g.

$$\mathrm{tr}(ABC) = \mathrm{tr}(BCA) = \mathrm{tr}(CAB).$$

Note, however, that this does *not* imply that e.g. $\mathrm{tr}(ABC) = \mathrm{tr}(ACB)$.

We can also define an **inner product** on the vector space of matrices:

The **Hilbert–Schmidt product** of A and B is given by

$$(A|B) = \frac{1}{2} \mathrm{tr} A^\dagger B.$$

The $\frac{1}{2}$ coefficient in this definition is simply the normalisation factor, which changes if we consider *multi-qubit* Pauli operators. It is not necessary, but simplifies some calculations.

We will return to the algebraic structure of these Pauli matrices in Chapter 7, before explaining how they turn out to be useful for things such as quantum error correction.

3.4 Unitaries as rotations

Now we can finish off our previous discussion (Section 2.10) of the Bloch sphere: we know how single-qubit state vectors correspond to points on the Bloch sphere, but now we can study how (2×2) unitary matrices correspond to *rotations* of this sphere.

Geometrically speaking, the group of (2×2) unitaries $\mathrm{U}(2)$ is a three-dimensional sphere S^3 in \mathbb{R}^4 . We often make the additional assumption that the determinant is equal to $+1$, and can then express these matrices as

$$U = u_0 \mathbf{1} + i(u_x \sigma_x + u_y \sigma_y + u_z \sigma_z).$$

Such matrices form a very important subgroup of $\mathrm{U}(2)$, called the **special** (meaning the determinant is equal to 1) unitary group, and denoted by $\mathrm{SU}(2)$.

In quantum theory, any two unitary matrices that differ by some global multiplicative phase factor represent the same physical operation, so we are “allowed to” fix the determinant to be $+1$, and thus restrict ourselves to considering matrices in $\mathrm{SU}(2)$. This is a sensible approach, practised by many theoretical physicists, but again, for some historical reasons, this convention is not usually followed in quantum information science. For example, phase gates are usually written as

$$P_\alpha = \begin{bmatrix} 1 & 0 \\ 0 & e^{i\alpha} \end{bmatrix}$$

rather than

$$P_\alpha = \begin{bmatrix} e^{-i\frac{\alpha}{2}} & 0 \\ 0 & e^{i\frac{\alpha}{2}} \end{bmatrix}$$

Still, as we’ve already mentioned, sometimes the T gate

$$T = \begin{bmatrix} 1 & 0 \\ 0 & e^{i\pi/4} \end{bmatrix} = \begin{bmatrix} e^{-i\pi/8} & 0 \\ 0 & e^{i\pi/8} \end{bmatrix}$$

is called the $\pi/8$ gate, because of its $\mathrm{SU}(2)$ form.

Here we’re going to work with $\mathrm{SU}(2)$, so that we can write any (2×2) unitary (i.e. up to an overall phase factor) as

$$U = u_0 \mathbf{1} + i(u_x \sigma_x + u_y \sigma_y + u_z \sigma_z) = u_0 \mathbf{1} + i\vec{u} \cdot \vec{\sigma}$$

where $u_0^2 + |\vec{u}|^2 = 1$.

This last restriction on u_0 and \vec{u} allows us to parametrise u_0 and \vec{u} in terms of a real unit vector \vec{n} , parallel to \vec{u} , and a real angle θ , in such a way that

$$U = (\cos \theta) \mathbf{1} + (i \sin \theta) \vec{n} \cdot \vec{\sigma}.$$

An alternative way of writing this expression is

$$U = e^{i\theta \vec{n} \cdot \vec{\sigma}},$$

as follows from the power-series expansion of the exponential. Indeed, any unitary matrix can always be written in the exponential form as

$$\begin{aligned} e^{iA} &= \mathbf{1} + iA + \frac{(iA)^2}{1 \cdot 2} + \frac{(iA)^3}{1 \cdot 2 \cdot 3} \cdots \\ &= \sum_{n=0}^{\infty} \frac{(iA)^n}{n!} \end{aligned}$$

where A is an anti-Hermitian matrix. This is analogous to writing complex numbers of unit modulus as $e^{i\alpha}$.

Now comes a remarkable connection between two-dimensional unitary matrices and ordinary three-dimensional rotations:

The unitary $U = e^{i\theta \vec{n} \cdot \vec{\sigma}}$ represents a clockwise rotation through the angle 2θ about the axis defined by \vec{n} .

The fact that the angle is 2θ , not θ , comes from our choice of parametrisation; the “better” convention is to parametrise so that $U = e^{i\frac{-\theta}{2} \vec{n} \cdot \vec{\sigma}}$, and then the direction follows from the right-hand rule, and the rotation corresponds to that in the Bloch sphere.

For example,

$$\begin{aligned} e^{i\theta \sigma_x} &= \begin{bmatrix} \cos \theta & i \sin \theta \\ i \sin \theta & \cos \theta \end{bmatrix} \\ e^{i\theta \sigma_y} &= \begin{bmatrix} \cos \theta & \sin \theta \\ -\sin \theta & \cos \theta \end{bmatrix} \\ e^{i\theta \sigma_z} &= \begin{bmatrix} e^{i\theta} & 0 \\ 0 & e^{-i\theta} \end{bmatrix} \end{aligned}$$

represent rotations by 2θ about the x -, y - and z -axis, respectively. In fact, these rotations are so important that they get a name.

Rotating a state about a **Pauli axis** (the x -, y -, or z -axes) is known as a **Pauli rotation**. We can write these as

$$e^{i\theta \sigma_k} = (\cos \theta) \mathbf{1} + (i \sin \theta) \sigma_k$$

for $k \in x, y, z$.

As you can see, we often make progress and gain insights simply by choosing a convenient parametrisation.

It is a good exercise to show that you can write any U in this way as well.

Be careful: the precise definition can vary a lot between different texts, with some including a factor of $1/2$, or even a negative sign.

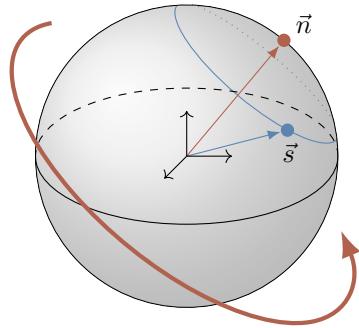


Figure 3.7: The matrix $e^{i\theta\vec{n}\cdot\vec{\sigma}}$ rotates the vector \vec{s} about \vec{n} by angle 2θ , sending it to a point on the blue circle, which is defined by being the unique circle whose centre is passed through by \vec{n} and containing \vec{s} .

Now we can show that the Hadamard gate

$$\begin{aligned} H &= \frac{1}{\sqrt{2}} \begin{bmatrix} 1 & 1 \\ 1 & -1 \end{bmatrix} \\ &= \frac{1}{\sqrt{2}}(\sigma_x + \sigma_z) \\ &= (-i)e^{i\frac{\pi}{2}\frac{1}{\sqrt{2}}(\sigma_x+\sigma_z)} \end{aligned}$$

represents (since we can ignore the global phase factor of $-i$) rotation about the diagonal $(x+z)$ -axis by an angle of π .

In somewhat abstract terms, we make the connection between unitaries and rotations by looking how the unitary group $U(2)$ acts on the three-dimensional vector space V of (2×2) Hermitian matrices *with zero trace*. All such matrices $S \in V$ can be written as $S = \vec{s} \cdot \vec{\sigma}$ for some real \vec{s} , i.e. each matrix is represented by a Euclidean vector \vec{s} in \mathbb{R}^3 .

Traceless matrices.

The vector space of **traceless matrices** (i.e. matrices S such that $\text{tr } S = 0$) might seem like an odd one, but it's actually one of the fundamental examples of a structure which is fundamental to modern mathematical physics, namely that of a **Lie algebra**. These arise when studying **Lie groups** — which are a combination of groups and manifolds, i.e. “a geometric space which has an algebraic structure” — via the notion of a **tangent space**.

In particular, the space of $(n \times n)$ traceless skew-Hermitian ($A^\dagger = -A$) matrices is the Lie algebra known as $\mathfrak{su}(2)$, which is the Lie algebra of $SU(2)$, since the latter is indeed a *Lie group*.

You might be wondering why we have suddenly switched to **skew-Hermitian** instead of **Hermitian**, but this is really just a mathematician/physicist convention: you can go from one to the other by simply multiplying by i . For example, mathematicians would usually prefer to work with $i\sigma_x$, $i\sigma_y$, and $i\sigma_z$ instead of the Pauli matrices σ_x , σ_y , and σ_z themselves; the former are skew-Hermitian, the latter are Hermitian.

Now, $U \in U(2)$ acts on the space V by $S \mapsto S' = USU^\dagger$, i.e.

$$\vec{s} \cdot \vec{\sigma} \longmapsto \vec{s}' \cdot \vec{\sigma} = U(\vec{s} \cdot \vec{\sigma})U^\dagger \quad (\ddagger)$$

This gives a linear map $\mathbb{R}^3 \rightarrow \mathbb{R}^3$, and is thus given by some (3×3) real-valued matrix:

$$R_U: \mathbb{R}^3 \rightarrow \mathbb{R}^3.$$

Next, note that this map is an **isometry** (a distance preserving operation), since it preserves the scalar product in the Euclidean space: for any two vectors \vec{s} and \vec{t} ,

$$\begin{aligned}\vec{s}' \cdot \vec{t}' &= \frac{1}{2} \text{tr}[S'T'] \\ &= \frac{1}{2} \text{tr}[(USU^\dagger)(UTU^\dagger)] \\ &= \frac{1}{2} \text{tr}[ST] \\ &= \vec{s} \cdot \vec{t}\end{aligned}$$

We will talk more about isometries in Section 9.3.

(where $S = \vec{s} \cdot \vec{\sigma}$ and $T = \vec{t} \cdot \vec{\sigma}$) using the cyclic property of the trace. This means that the matrix R_U is **orthogonal**: orthogonal transformations preserve the length of vectors as well as the angles between them.

Furthermore, we can show that $\det R_U = 1$. But the *only* isometries in three dimensional Euclidean space (which are described by orthogonal matrices with determinant 1) are rotations.

Thus, in the mathematical lingo, we have established a group homomorphism

$$\begin{aligned}\text{U}(2) &\longrightarrow \text{SO}(3) \\ U &\longmapsto R_U\end{aligned}$$

where $\text{SO}(3)$ stands for the **special orthogonal group** in three dimensions — the group of all rotations about the origin of three-dimensional Euclidean space \mathbb{R}^3 under the operation of composition, which can be represented by the group of (3×3) orthogonal (and thus *real*) matrices. It follows from Equation (‡) that unitary matrices differing only by a global multiplicative phase factor (e.g. U and $e^{i\varphi}U$) represent the same rotation.

Some mathematicians might say that $\det R_U = 1$ because “any matrix in $\text{U}(2)$ can be smoothly connected to the identity”.

Recall that a **homomorphism** is a structure-preserving map between two algebraic structures of the same type; in our case, two groups. An **isomorphism** between algebraic structures of the same type is a homomorphism that has an inverse homomorphism.

Vectors.

This mathematical argument is secretly using the language of unit **quaternions**, also known as **versors**, since these provide a very convenient way of describing **spatial rotation**, and are often used in e.g. 3D computer graphics software.

Physicists, however, usually prefer a more direct demonstration of this rotation interpretation, which might go roughly as follows. Consider the map $\vec{s} \mapsto \vec{s}'$ induced by $U = e^{i\alpha\vec{n} \cdot \vec{\sigma}}$. For small values of α , we can write

$$\begin{aligned}\vec{s}' \cdot \vec{\sigma} &= U(\vec{s} \cdot \vec{\sigma})U^\dagger \\ &= (\mathbf{1} + i\alpha(\vec{n} \cdot \vec{\sigma}) + \dots)(\vec{s} \cdot \vec{\sigma})(\mathbf{1} - i\alpha(\vec{n} \cdot \vec{\sigma}) + \dots).\end{aligned}$$

To the first order in α , this gives

$$\vec{s}' \cdot \vec{\sigma} = (\vec{s} + 2\alpha(\vec{n} \times \vec{s})) \cdot \vec{\sigma}$$

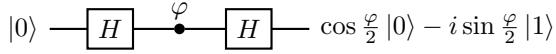
that is,

$$\vec{s}' = \vec{s} + 2\alpha(\vec{n} \times \vec{s})$$

which we recognise as a good old textbook formula for an infinitesimal clockwise rotation of \vec{s} about the axis \vec{n} through the angle 2α .

3.5 Universality, again

Although this may all seem tediously abstract, it is surprisingly useful. Take another look at the single-qubit interference circuit

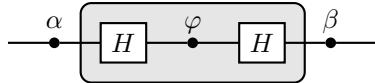


and the corresponding sequence of unitary operations

$$\begin{aligned} H \left(e^{-i\frac{\varphi}{2}Z} \right) H &= e^{-i\frac{\varphi}{2}X} \\ &= \begin{bmatrix} \cos \varphi/2 & -i \sin \varphi/2 \\ -i \sin \varphi/2 & \cos \varphi/2 \end{bmatrix} \end{aligned}$$

The single-qubit interference circuit has a simple geometrical meaning: it shows how a rotation about the z -axis, induced by the phase gate P_φ , is turned, by the two Hadamard gates, into a rotation about the x -axis.

Now, take a look at this circuit:



What does it represent? The central part is a rotation by φ about the x -axis, sandwiched between two rotations about the z -axis. Recall our previous discussion (Section 2.12) about a universal set of gates: any rotation in the Euclidean space can be performed as a sequence of three rotations: one about z -axis, one about x -axis, and one more about the z -axis. In this context, this implies that any unitary U , up to a global phase factor, can be written as

$$\begin{aligned} U(\alpha, \beta, \varphi) &= e^{-i\frac{\beta}{2}Z} e^{-i\frac{\varphi}{2}X} e^{-i\frac{\alpha}{2}Z} \\ &= \begin{bmatrix} e^{-i(\frac{\alpha+\beta}{2})} \cos \frac{\varphi}{2} & ie^{i(\frac{\alpha-\beta}{2})} \sin \frac{\varphi}{2} \\ ie^{-i(\frac{\alpha-\beta}{2})} \sin \frac{\varphi}{2} & e^{i(\frac{\alpha+\beta}{2})} \cos \frac{\varphi}{2} \end{bmatrix}. \end{aligned}$$

That is, once you are given a pair of Hadamard gates and an infinite supply of phase gates (so that you can choose the three phases you need) you can construct an *arbitrary* unitary operation on a single qubit.

It is important to note that the two axes in question, z and x , do not have any special status, geometrically speaking — if we have rotations about any two orthogonal axes then we can create any one-qubit unitary that we want.

In fact, even this orthogonality condition isn't necessary! See Figure 3.8

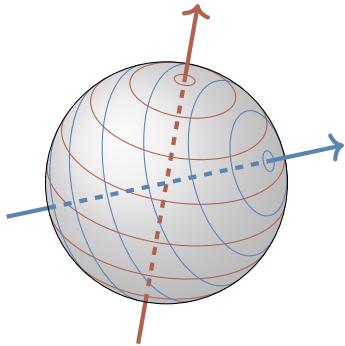
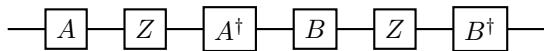


Figure 3.8: If we can move along the two families of circles, then from any point on the sphere we can reach any other point. The two axes do not even have to be orthogonal: any two different (i.e. non-collinear) axes will do! Can you see why?

Now consider the following circuit:



where both A and B are unitary operations. We claim that *any* unitary U can be represented in this form, for some A and B .

Again, we can prove this geometrically. The circuit represents two rotations by 180° about the two axes obtained by rotating the z -axis via unitaries A and B , respectively. Any rotation in the three-dimensional space is the composition of two rotations by 180° , as shown in Figure 3.9. The resulting axis of rotation is perpendicular to the two axes about which rotations by 180° are performed, and the angle of the composed rotation is twice the angle between the two axes.

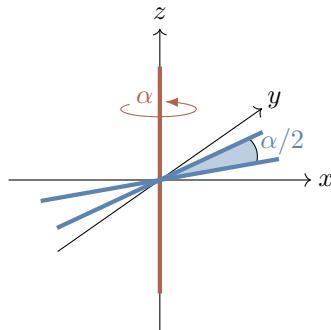


Figure 3.9: Rotating by α around the z -axis is the same as the composition of two rotations by 180° around axes which both lie in the xy -plane, with angle $\alpha/2$ between them.

3.6 Some quantum dynamics

We will finish this chapter with a short aside on some more fundamental quantum theory. Although this isn't our main focus — we will happily black box away this stuff, happy in the knowledge that some scientists in a lab somewhere have already packaged everything up into nice quantum logic gates that we can use — it is a nice opportunity to talk about other aspects of the subject that might be of interest.

The time evolution of a quantum state is a unitary process which is generated by a Hermitian operator called the **Hamiltonian**, which we denote by \hat{H} . The Hamiltonian contains a complete specification of all interactions within the system under

consideration — in general, it may change over time. In an isolated system, the state vector $|\psi(t)\rangle$ changes smoothly in time according to the **Schrödinger equation**:

$$\frac{d}{dt}|\psi(t)\rangle = -\frac{i}{\hbar}\hat{H}|\psi(t)\rangle.$$

In the same way that [Newton's second law](#) describes certain future behaviour of a classical system given some initial knowledge, Schrödinger's equation describes the future behaviour of a quantum system given some initial knowledge.

Lagrangian and Hamiltonian mechanics.

The first approach towards classical mechanics that you might meet is the **Newtonian** framework, where we talk about the equations that are satisfied by **forces**. It is Newton's second law that we usually apply the most in order to describe the behaviour of classical systems, and it is usually stated as $\mathbf{F} = m\mathbf{a}$, where m is mass and \mathbf{a} is acceleration. But really the notion of “force” is not a fundamental one — a slightly more instructive way of writing Newton's second law for a system whose mass can change over time is as $\mathbf{F} = \frac{d\mathbf{p}}{dt}$, where $\mathbf{p} = m\mathbf{v}$ is (linear) **momentum**: the product of mass (a scalar) with velocity (a vector).

Instead of talking about *forces* within a system, we can instead describe things entirely in terms of either *position and velocity* (where the latter is just the time derivative of the former) — using coordinates $(\mathbf{q}, \dot{\mathbf{q}})$, where \mathbf{q} (confusingly) stands for “position”, and we write $\dot{\mathbf{q}}$ to mean $\frac{d}{dt}\mathbf{q}$ — or *position and momentum* — using coordinates (\mathbf{q}, \mathbf{p}) , where (again, confusingly) \mathbf{p} stands for momentum (maybe it's like “pneumatic”, and we should call it “pmomentum”).

If we take either of these two approaches, then we have a suitable replacement for Newton's second law:

1. The first approach results in [Lagrangian mechanics](#), where we have some function $\mathcal{L}(t, \mathbf{q}(t), \dot{\mathbf{q}}(t))$ called the **Lagrangian**, and study the [Euler–Lagrange equations](#)

$$\frac{d}{dt} \left(\frac{\partial \mathcal{L}}{\partial \dot{\mathbf{q}}} \right) = \frac{\partial \mathcal{L}}{\partial \mathbf{q}}$$

which is a second-order differential equation.

2. The second approach results in [\[Hamiltonian mechanics\]](#), where we have some function $\mathcal{H}(t, \mathbf{p}(t), \mathbf{q}(t))$ called the **Hamiltonian**, and study the **Hamilton equations**

$$\begin{aligned} \frac{d\mathbf{q}}{dt} &= \frac{\partial \mathcal{H}}{\partial \mathbf{p}} \\ \frac{d\mathbf{p}}{dt} &= -\frac{\partial \mathcal{H}}{\partial \mathbf{q}} \end{aligned}$$

which is a pair of first-order differential equations.

These two important functions, the Lagrangian and the Hamiltonian, are given by the total energies of the system: the former is the *difference* of the kinetic and potential energies; the latter is the *sum* of the kinetic and potential energies.

There are many situations where one framework is more useful than the other, but in quantum physics we normally find the Hamiltonian approach more easier than the Lagrangian, since momentum is a conserved quantity, whereas velocity is not. In fact, the Hamiltonian approach is hidden all over

the place: the position and momentum operators in quantum physics are truly fundamental, and will show up again when we talk about **uncertainty principles** in Section 4.6.

Here \hbar is a very (*very*) small number known as the **Planck constant**. Physicists often pick a unit system such that \hbar is equal to 1, to make calculations simpler. But in SI units, $2\pi\hbar$ is *exactly* equal to $6.62607015 \times 10^{-34}$ joules per hertz.

As a historical note, Planck's constant \hbar has its roots right in the very birth of quantum physics, since it shows up in the equation for the energy of a photon. More generally, in 1923 [de Broglie](#) postulated that the ratio between the momentum and quantum wavelength of *any* particle would be $2\pi\hbar$. Even before this, it turned up in 1905 when Einstein stated his support for Planck's idea that light is not just a wave, but simultaneously consists of tiny packets of energy, called **quanta** (whence the name quantum physics!), which we now call electrons. We will see the Planck constant turn up again when we talk about **uncertainty principles** in Section 4.6.

Back to quantum dynamics. For **time-independent** Hamiltonians $\hat{H}(t) = \hat{H}$, the formal solution of the Schrödinger equation is given by

$$|\psi(t)\rangle = e^{-\frac{i}{\hbar}\hat{H}t}|\psi(0)\rangle.$$

Note that the function $|\psi(t)\rangle$ thus obtained is **separable**: it is written as a product of two functions $e^{-\frac{i}{\hbar}\hat{H}t} \cdot |\psi(0)\rangle$, where the first is purely time dependent, and the second has no time dependence. In fact, the time-dependent part is exactly a phase factor $U(t) = e^{-\frac{i}{\hbar}\hat{H}t}$, and we know that this does not affect the resulting probabilities: $||\psi(t)\rangle|^2 = |U(t)|^2||\psi(0)\rangle|^2 = ||\psi(0)\rangle|^2$. This means that $||\psi(t)\rangle|^2$ is constant throughout time — we call such a state **stationary**, or refer to it as a **standing wave**.

The kilogram is now *defined* in SI in terms of the Planck constant, the speed of light, and the atomic transition frequency of of caesium-133.

The whole history of quantum physics, arguably starting with the **black-body problem**, accounting for the [Rayleigh-Jeans law](#), and leading on to the discovery of the [photoelectric effect](#), is a wonderful story, but one that we do not have the space to tell here.

Quantum confinement.

We will *not* delve into a proper study of the Schrödinger equation — this is the subject of entire books already, and deserves a lengthy treatment — but it is nice to mention at least one worked example (although we will skip almost all of the details!), since its applications are commonplace in day-to-day life.

In the time-independent case, the Schrödinger equation can simply be written as $\hat{H}|\psi\rangle = E|\psi\rangle$, where E is the total energy of the system. When written like this, we can sneak a glimpse at what the Hamiltonian is really all about: it is some operator whose eigenstates are solutions of the Schrödinger equation, and whose eigenvalues are the corresponding energy levels.

One particularly instructive situation to consider is that of [a particle in a box](#): we have some 1-dimensional region of space in which a particle is free to move around, but outside of this finite segment there is infinite potential energy, restricting the particle from moving beyond this region. It turns out that, in this case, the Hamiltonian is given by

$$\hat{H} = -\frac{\hbar^2}{2m} \frac{d^2}{dx^2}$$

and the general solution to the resulting Schrödinger equation can be shown to be

$$\psi(x) = C \sin(kx) + D \cos(kx)$$

where $k = n\pi/L$ for some positive integer n , and where L is the length of the potential-free region. This implies (after some algebra) that the energy

$E = E_n$ of the solution with $k = n\pi/L$ is equal to

$$E_n = \frac{(2\pi n\hbar)^2}{8mL^2}.$$

What is utterly unique and important to quantum physics is not really this specific fraction, but the fact that *the possible energy levels of the system are purely discrete* — energy cannot be any real value, as is the case in the classical world, but it can only take values within some discrete set $\{E_1, E_2, \dots\}$.

But what are the applications of this particle in a box? Well, this phenomena of a system having a *discrete* energy spectrum when restrained to small enough spaces is known as **quantum confinement**, and **quantum well lasers** are laser diodes which have a small enough active region for this confinement to occur. Such lasers are arguably the most important component of fiber optic communications, which form the underlying foundations of the internet itself.

Before moving on to understand the relevance of this to what we have already been discussing, let us take a moment to see why we might have expected to stumble across such a solution as $e^{-\frac{i}{\hbar}\hat{H}t}$ (or, from the opposite point of view, how we could derive the Schrödinger equation). We start with state vectors, which we want to evolve according to transition operators — we have already justified why we should think about representing these transitions by matrices (namely because matrices simply package up all the multiplication and addition in the “right” way). But now we want these evolutions to be *continuous*, whatever that might formally mean.

For a start, this means that we want not only to be able to multiply the matrices that represent these transitions, but also to do the inverse: take any transition and “chop it up” into smaller time chunks, viewing any evolution T as a sequence $T_n T_{n-1} \dots T_1$ of evolutions T_i that take place on a shorter time scale. Directly then, we want to be able to consider *roots* (square roots, cube roots, and so on) of our matrices, which means that they must at the very least have complex entries.

But let us try to take this continuity requirement a bit more seriously: say that any transition T is parametrised by a real parameter t , which we will think of as “time”. It makes sense to ask for $T(t+t') = T(t)T(t')$ for any $t, t' \in \mathbb{R}$, and to say that “at time 0, things are exactly how we found them”, i.e. $T(0) = \mathbf{1}$. But we know how to solve for such requirements: take $T(t) = \exp(tX)$, where X is an arbitrary complex matrix! This also solves the problem of wanting to take roots, since $T(t)^{\frac{1}{n}} = T(t/n)$, and $T(t)^{-1} = T(-t)$.

Next, as we’ve already mentioned, complex matrices have a polar form — analogous to how any $z \in \mathbb{C}$ can be written as $z = re^{i\varphi}$, we can write any complex matrix Z as $Z = RU$, where R is positive semi-definite and U is unitary. In this decomposition, just as for the polar decomposition $z = re^{i\varphi}$, the R corresponds to “stretching” and the U corresponds to “rotation”. But we don’t want to have to worry about convergence issues, and the idea of “exponential stretching” sounds like it might give us some problems, so let us just consider $Z = RU$ with $R = \mathbf{1}$, i.e. just *unitary* matrices. And if we want $T(t)$ to be unitary, then it suffices to take X to be anti-Hermitian.

In summary, from just asking for our evolutions to be continuous and not have any convergence issues, we end up with the conclusion that we are interested in evolutions described by exponentials of anti-Hermitian matrices, i.e. $U(t) = \exp(itX)$ for some Hermitian matrix X .

Stone's theorem.

This correspondence between so-called **one-parameter unitary groups** — families $(U_t)_{t \in \mathbb{R}}$ of unitary operators (satisfying some analytic property) — and Hermitian operators, given by $U_t = e^{itA}$, is known as **Stone's theorem (on one-parameter unitary groups)**.

For example, if we consider the **translation** operators T_t , which are defined by $T_t(\psi)(x) = \psi(x + t)$, then we have the corresponding Hermitian operator $-i\frac{d}{dx}$, which is known (for good reason) as the **momentum operator**. In fancy words, this says that *1-dimensional motion is infinitesimally generated by momentum*.

Now, to relate this to the earlier parts of this chapter, we note that the Hamiltonian of a qubit can always be written in the form $H = E_0 \mathbf{1} + \omega(\vec{n} \cdot \vec{\sigma})$, hence

$$\begin{aligned} U(t) &= e^{-i\omega t \vec{n} \cdot \vec{\sigma}} \\ &= (\cos \omega t) \mathbf{1} - (i \sin \omega t) \vec{n} \cdot \vec{\sigma} \end{aligned}$$

which is a rotation with angular frequency ω about the axis defined by the unit vector \vec{n} .

The 4π world of qubits.

This section is not yet finished.

3.7 Remarks and exercises

3.7.1 Quantum bomb tester

You have been drafted by the government to help in the demining effort in a former war-zone. In particular, retreating forces have left very sensitive bombs in some of the sealed rooms. The bombs are configured such that if even one photon of light is absorbed by the fuse (i.e. if someone looks into the room), the bomb will go off. Each room has an input and output port which can be hooked up to external devices. An empty room will let light go from the input to the output ports unaffected, whilst a room with a bomb will explode if light is shone into the input port and the bomb absorbs even just one photon — see Figure 3.10.

This is a slightly modified version of a bomb-testing problem described by Avshalom Elitzur and Lev Vaidman in *Quantum-mechanical interaction-free measurement*, Found. Phys. 47 (1993), pp. 987–997.

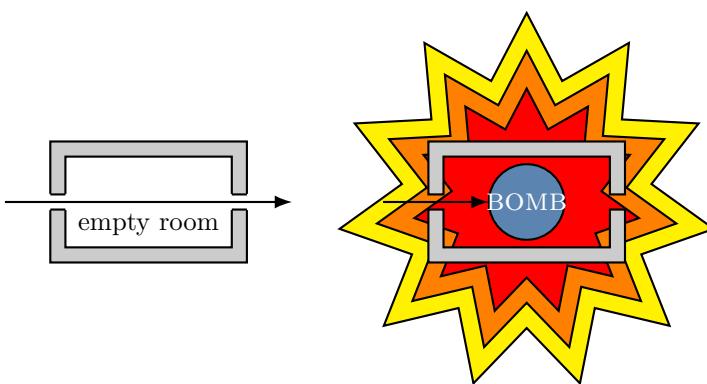


Figure 3.10: *Left:* the passage of a photon through an empty room. *Right:* the passage of a photon through a room containing a bomb.

Your task is to find a way of determining whether a room has a bomb in it without blowing it up, so that specialised (limited and expensive) equipment can be devoted

to defusing that particular room. You would like to know whether a particular room has a bomb in it with as much certainty as possible.

1. To start with, consider the setup in Figure 3.11, where the input and output ports are hooked up in the lower arm of a Mach-Zehnder interferometer.
 - a. Assume an empty room. Send a photon to input port $|0\rangle$. Which detector, at the output port, will register the photon?
 - b. Now assume that the room does contain a bomb. Again, send a photon to input port $|0\rangle$. Which detector will register the photon and with which probability?
 - c. Design a scheme that allows you — at least some of the time — to decide whether a room has a bomb in it without blowing it up. If you iterate the procedure, what is its overall success rate for the detection of a bomb without blowing it up?

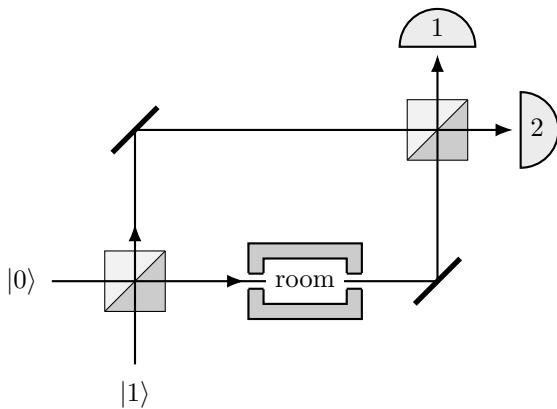


Figure 3.11: The Mach-Zehnder interferometer hooked up to the bomb-testing room.

2. Assume that the two beam splitters in the interferometer are different. Say the first beam-splitter reflects incoming light with probability R and transmits with probability $T = 1 - R$, but the second one transmits with probability R and reflects with probability T (that is, the two beam-splitters are *asymmetric*, but “inverse” to one another). Would the new setup improve the overall success rate of the detection of a bomb without blowing it up?
3. There exists a scheme, involving many beam-splitters and something called the **quantum Zeno effect**, such that the success rate for detecting a bomb without blowing it up approaches 100%. Try to work it out, or find a solution on the internet.

You can play around with this setup on the [Quantum Flytrap Virtual Lab](#).

3.7.2 Orthonormal Pauli basis

Show that $\{1, \sigma_x, \sigma_y, \sigma_z\}$ is an orthonormal basis of the space of complex (2×2) matrices with respect to the Hilbert-Schmidt product.

3.7.3 Pauli matrix expansion coefficients

Recall that any (2×2) complex matrix A has a unique expansion in the form

$$\begin{aligned} A &= \begin{bmatrix} a_0 + a_z & a_x - ia_y \\ a_x + ia_y & a_0 - a_z \end{bmatrix} \\ &= a_0 \mathbf{1} + a_x \sigma_x + a_y \sigma_y + a_z \sigma_z \\ &= a_0 \mathbf{1} + \vec{a} \cdot \vec{\sigma}. \end{aligned} \tag{*}$$

for some complex numbers a_0, a_x, a_y , and a_z .

1. Show that the coefficients a_k (for $k = x, y, z$) are given by the inner product $a_k = (\sigma_k | A) = \frac{1}{2} \text{tr } \sigma_k A$.

In these notes, we usually deal with matrices that are Hermitian ($A = A^\dagger$) or unitary ($AA^\dagger = \mathbf{1}$). It is easy to see that, if A is Hermitian, then a_0 and the three components of \vec{a} are all real. The (2×2) unitaries are usually parametrised as

$$U = e^{i\varphi} \left(u_0 \mathbf{1} + i(u_x \sigma_x + u_y \sigma_y + u_z \sigma_z) \right)$$

where $e^{i\varphi}$ is an overall multiplicative phase factor, with φ real, and u_0 and the three components u_x, u_y, u_z are all real numbers.

2. Show that the unitarity condition implies that

$$u_0^2 + u_x^2 + u_y^2 + u_z^2 = 1$$

and show, using this parametrisation, that the determinant of U is $e^{i2\varphi}$.

3.7.4 Linear algebra of the Pauli vector

In what follows, we use the notation from our algebraic treatment of Pauli operators in Section 3.3, where we defined the **Pauli vector** $\vec{\sigma}$.

1. Show that $\frac{1}{2} \text{tr}(\vec{a} \cdot \vec{\sigma})(\vec{b} \cdot \vec{\sigma}) = \vec{a} \cdot \vec{b}$.
2. Show that any $\vec{n} \cdot \vec{\sigma}$ has eigenvalues $\pm |\vec{n}|$.
3. Show that, if $\vec{n} \cdot \vec{m} = 0$, then the operators $\vec{n} \cdot \vec{\sigma}$ and $\vec{m} \cdot \vec{\sigma}$ anticommute.

Hint: you may find Exercise 2.14.9 helpful.

3.7.5 Matrix Euler formula

1. Show that, if $A^2 = \mathbf{1}$, then we can manipulate the power series expansion of e^{iA} into a simple expression: for any real α ,

$$e^{i\alpha A} = (\cos \alpha) \mathbf{1} + (i \sin \alpha) A.$$

2. Show that any (2×2) unitary matrix U can be written, up to an overall multiplicative phase factor, as

$$U = e^{i\theta \vec{n} \cdot \vec{\sigma}} = (\cos \theta) \mathbf{1} + (i \sin \theta) \vec{n} \cdot \vec{\sigma}.$$

Hint: the argument here is the same as the argument that $e^{i\theta} = \cos \theta + i \sin \theta$.

3.7.6 Special orthogonal matrix calculations

1. Show that $\text{tr } \sigma_x \sigma_y \sigma_z = 2i$.
2. Let U be a unitary matrix, and write \vec{e}_x , \vec{e}_y , and \vec{e}_z to mean the unit vectors along the x -, y -, and z -axis, respectively. We define new unit vectors \vec{f}_x , \vec{f}_y , and \vec{f}_z by applying U to our existing unit vectors. Then

$$U(\vec{e}_k \cdot \sigma_k)U^\dagger = U\sigma_k U^\dagger = \vec{f}_k \cdot \vec{\sigma}.$$

We already know that, in Euclidean space, this transformation is described by a (3×3) orthogonal matrix R_U . How are the three vectors \vec{f}_x , \vec{f}_y , and \vec{f}_z related to the entries in matrix R_U ?

3. Show that

$$\begin{aligned}\text{tr } \sigma_x \sigma_y \sigma_z &= \text{tr}(\vec{f}_x \cdot \vec{\sigma})(\vec{f}_y \cdot \vec{\sigma})(\vec{f}_z \cdot \vec{\sigma}) \\ &= 2i \det R_U\end{aligned}$$

(which implies that $\det R_U = 1$).

4. Use the orthonormality of the Pauli basis along with Equation (‡) to show that the elements of the matrix $R = R_U$ can be expressed in terms of those of the matrix U , in the form

$$R_{ij} = \frac{1}{2} \text{tr} (\sigma_i U \sigma_j U^\dagger).$$

Here, i and j take values in $\{1, 2, 3\}$, and $\sigma_1 \equiv \sigma_x$, $\sigma_2 \equiv \sigma_y$, $\sigma_3 \equiv \sigma_z$.

3.7.7 Phase as rotation

1. Show that the phase gate

$$P_\varphi = \begin{bmatrix} 1 & 0 \\ 0 & e^{i\varphi} \end{bmatrix}$$

represents an anticlockwise rotation about the z -axis through the angle φ .

Hint: it might be helpful to start with the SU(2) version of the phase gate:

$$\begin{aligned}P_\varphi &= e^{-i\frac{\varphi}{2}\sigma_z} \\ &= \begin{bmatrix} e^{-i\frac{\varphi}{2}} & 0 \\ 0 & e^{i\frac{\varphi}{2}} \end{bmatrix}\end{aligned}$$

which gives

$$R_{P_\varphi} = \begin{bmatrix} \cos \varphi & -\sin \varphi & 0 \\ \sin \varphi & \cos \varphi & 0 \\ 0 & 0 & 1 \end{bmatrix}$$

Hint: recall Section 3.5.

3.7.8 Calculating a Pauli rotation

1. Express the Pauli rotation $e^{i\sigma_y\pi/3}$ as a matrix.
2. Give a decomposition of this rotation in the form

$$R_Z(\alpha)HR_Z(\beta)HR_Z(\gamma)$$

where $R_Z(\theta)$ denotes a Pauli σ_z -rotation by angle θ .

3.7.9 Geometry of the Hadamard

1. Express the Hadamard gate H in terms of $\vec{n} \cdot \vec{\sigma}$, and show that

$$HZH = X$$

$$HXH = Z$$

$$HYH = -Y.$$

2. Show that the Hadamard gate H turns rotations about the x -axis into rotations about the z -axis, and vice versa. That is,

$$H \left(e^{-i\frac{\varphi}{2}Z} \right) H = e^{-i\frac{\varphi}{2}X}$$

$$H \left(e^{-i\frac{\varphi}{2}X} \right) H = e^{-i\frac{\varphi}{2}Z}.$$

3.7.10 Swiss Granite Fountain

In the Singapore Botanic Gardens, there is a sculpture by Ueli Fausch called “Swiss Granite Fountain”. It is a spherical granite ball which measures 80cm in diameter and weighs 700kg, and is kept afloat by strong water pressure directed through a basal block. It is easy to set the ball in motion, and it keeps rotating in whatever way you start for a long time. Suppose you are given access to this ball only near the top, so that you can push it to make it rotate around any horizontal axis, but you don’t have enough of a grip to make it turn around the vertical axis. Can you make it rotate around the vertical axis anyway?

3.7.11 Dynamics in a magnetic field

A qubit initially in state $|0\rangle$ is placed in a uniform magnetic field. The interaction between the field and the qubit is described by the Hamiltonian

$$H = \omega \begin{bmatrix} 0 & -i \\ i & 0 \end{bmatrix}$$

where ω is proportional to the strength of the field. What is the state of the qubit after time $t = \pi/4\omega$?

In Earth's magnetic field, which is about 0.5 gauss, the value of ω is of the order of 10^6 cycles per second.

4 Measurements

About the Hilbert-space formalism of quantum theory, and the role of measurements in quantum information theory, as well as introducing the quantum dramas of Alice and Bob.

Eventually we have to talk about **quantum measurements**, since, at some point, someone has to look at a measuring device and register the outcome of whatever quantum circuits we've been designing. It turns out that this is a bit more tricky than one might think. Quantum measurement is not a passive acquisition of information: if you measure, you disturb. Even though it is a physical process, like any other quantum evolution, it is traditionally described by a different set of mathematical tools.

4.1 Hilbert spaces, briefly

A formal mathematical setting for a quantum system is that of a **Hilbert space** \mathcal{H} , which is (for us) just a vector space along with an inner product.

Given a Hilbert space corresponding to our system, the result of any preparation of the system is then represented by some *unit* vector $|\psi\rangle \in \mathcal{H}$, and any test is represented by some other unit vector $|e\rangle \in \mathcal{H}$. The inner product of these two vectors, $\langle e|\psi\rangle$, gives the probability amplitude that an object prepared in state $|\psi\rangle$ will pass a test for being in state $|e\rangle$. As always, probabilities are obtained by squaring absolute values of probability amplitudes:

$$|\langle e|\psi\rangle|^2 = \langle\psi|e\rangle\langle e|\psi\rangle.$$

After the test, in which the object was found to be in state $|e\rangle$, say, the object forgets about its previous state $|\psi\rangle$ and is, indeed, actually now in state $|e\rangle$. That is, if we immediately measure the object again, we will find it to still be in state $|e\rangle$ with probability 1. This is the mysterious **quantum collapse**, which we will further discuss later on.

A more complete test involves multiple states e_k that form an orthonormal basis $\{|e_1\rangle, \dots, |e_n\rangle\}$. These states are perfectly distinguishable from each other: the condition $\langle e_k|e_l\rangle = \delta_{kl}$ implies that a quantum system prepared in state $|e_l\rangle$ will never be found in state $|e_k\rangle$ (unless $k = l$). The probability amplitude that the system in state $|\psi\rangle$ will be found in state $|e_k\rangle$ is $\langle e_k|\psi\rangle$ and, given that the vectors $|e_k\rangle$ span the whole vector space, the system will be always found in one of the basis states, whence

$$\sum_k |\langle e_k|\psi\rangle|^2 = 1.$$

As a result:

A **complete** measurement in quantum theory is determined by the choice of an orthonormal basis $\{|e_i\rangle\}$ in \mathcal{H} , and every such basis (in principle) represents a possible complete measurement.

4.2 Complete measurements

A **projector** is any Hermitian ($P = P^\dagger$) operator which is **idempotent** ($P^2 = P$). The **rank** of P is given by $\text{tr}(P)$. In the Dirac notation, if $|e\rangle$ is a unit vector, then $|e\rangle\langle e|$ is a rank-one projector on the subspace spanned by $|e\rangle$, and it acts on any vector $|v\rangle$ via $(|e\rangle\langle e|)|v\rangle = |e\rangle\langle e|v\rangle$.

As mentioned in Section 0.3, we only work with *finite dimensional* vector spaces, and it is a very convenient fact that any finite dimensional inner product space is automatically a Hilbert space.

The most common measurement in quantum information science is the **standard measurement** on a qubit, also referred to as the measurement in the **standard (or computational) basis**: $\{|0\rangle, |1\rangle\}$. When we draw circuit diagrams it is tacitly assumed that such a measurement is performed on each qubit at the end of quantum evolution.

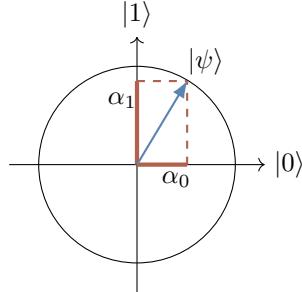


Figure 4.1: The standard/computational basis defines the so-called standard measurements.

However, if we want to emphasise the role of the measurement, then we can include it explicitly in the diagram as a special quantum gate, e.g. as

$$|\psi\rangle = \alpha_0 |0\rangle + \alpha_1 |1\rangle \xrightarrow{\text{gate}} \begin{cases} |0\rangle & \text{with probability } |\alpha_0|^2 \\ |1\rangle & \text{with probability } |\alpha_1|^2 \end{cases}$$

or, in an alternative notation, as

$$|\psi\rangle = \alpha_0 |0\rangle + \alpha_1 |1\rangle \xrightarrow{k} |k\rangle \quad \text{with probability } |\alpha_k|^2 \quad (k = 0, 1)$$

As we can see, if the qubit is prepared in state $|\psi\rangle = \alpha_0|0\rangle + \alpha_1|1\rangle$ and subsequently measured in the standard basis, then the outcome is $|k\rangle$ (for $k = 0, 1$) with probability

$$\begin{aligned} |\alpha_k|^2 &= |\langle k|\psi\rangle|^2 \\ &= \underbrace{\langle \psi|}_{{\alpha_k^*}} \underbrace{|k\rangle}_{\alpha_k} \langle k|\psi\rangle \\ &= \langle \psi| \underbrace{|k\rangle\langle k|}_{\text{projector}} |\psi\rangle \\ &= \langle \psi| P_k |\psi\rangle \end{aligned}$$

This slick argument is a good example of how nice the bra-ket notation can be when we leverage the ambiguity of an expression like $\langle a||b\rangle|b\rangle\langle a|$, which we can read as the scalar product of two scalars or as a projector sandwiched between a bra and a ket.

where $P_k = |k\rangle\langle k|$ is the projector on $|k\rangle$. If the outcome of the measurement is $|k\rangle$, then the output state of the measurement gate is $|k\rangle$. The original state $|\psi\rangle$ is *irretrievably lost*. This sudden change of the state, from the pre-measurement state $|\psi\rangle$ to the post-measurement state, either $|0\rangle$ or $|1\rangle$, is often called a **collapse** or a **reduction** of the state.

So it looks like there are two distinct ways for a quantum state to change: on the one hand we have unitary evolutions, and on the other hand we have an abrupt change during the measurement process. Surely, the measurement process is not governed by any different laws of physics?

No, it is not!

Quantum collapse.

The subtleties (both mathematical and philosophical) of quantum collapse are still very much active topics of research, and we could spend an entire book discussing them. There are a *lot* of other sources where you can read about such things — here is a very short list to start:

- T. Norson, *Foundations of Quantum Mechanics: An Exploration of the Physical Meaning of Quantum Theory*. Springer, 2017. ISBN: 978-3-319-65867-4. DOI: [10.1007/978-3-319-65867-4](https://doi.org/10.1007/978-3-319-65867-4).
- M. Schlosshauer, “Decoherence, the measurement problem, and interpretations of quantum mechanics”. Rev. Mod. Phys. 76 (2004), pp. 1267–1305. arXiv:[quant-ph/0312059](https://arxiv.org/abs/quant-ph/0312059).
- F. Giacosa, “On unitary evolution and collapse in Quantum Mechanics”. Quanta 3 (2014), pp. 156–170. arXiv:[1406.2344](https://arxiv.org/abs/1406.2344).

A measurement is a physical process and can be explained without any “collapse”, but it is usually a complicated process in which one complex system (a measuring apparatus or an observer) interacts and gets correlated with a physical system being measured. We will discuss this more later on, but for now let us accept a “collapse” as a *convenient mathematical shortcut*, and describe it in terms of projectors rather than unitary operators.

For our purposes, the idea of “quantum collapse” is simply a way of black boxing the *irreversible* interaction between a quantum system and its surrounding classical environment.

On a practical level, it means that we describe measurement and observation with projectors instead of unitary operators.

4.3 The projection rule, and incomplete measurements

So far we have identified measurements with orthonormal bases, or, if you wish, with a set of orthonormal projectors on the basis vectors.

An orthonormal basis satisfies two conditions:

- **Orthonormality:** $\langle e_k | e_l \rangle = \delta_{kl}$
- **Completeness:** $\sum_k |e_k\rangle\langle e_k| = \mathbf{1}$

Given a quantum system in state $|\psi\rangle$ such that $|\psi\rangle = \sum_k \alpha_k |e_k\rangle$, we can write

$$\begin{aligned} |\psi\rangle &= \mathbf{1}|\psi\rangle \\ &= \sum_k (|e_k\rangle\langle e_k|)|\psi\rangle \\ &= \sum_k |e_k\rangle\langle e_k|\psi\rangle \\ &= \sum_k |e_k\rangle\alpha_k \\ &= \sum_k \alpha_k |e_k\rangle \end{aligned}$$

which tells us that *any* vector in \mathcal{H} can be expressed as the sum of the orthogonal pro-

jections on the $|e_k\rangle$, whence the name of the “completeness” condition. This says that the measurement in the basis $\{|e_i\rangle\}$ gives the outcome labelled by e_k with probability

$$|\langle e_k | \psi \rangle|^2 = \langle \psi | e_k \rangle \langle e_k | \psi \rangle$$

and leaves the system in state $|e_k\rangle$. This is a *complete* measurement, which represents the best we can do in terms of resolving state vectors in the basis states. But sometimes we do not want our measurement to distinguish *all* the elements of an orthonormal basis.

For example, a complete measurement in a four-dimensional Hilbert space will have four distinct outcomes: $|e_1\rangle$, $|e_2\rangle$, $|e_3\rangle$, and $|e_4\rangle$, but we may want to lump together some of the outcomes and distinguish, say, only between $\{|e_1\rangle, |e_2\rangle\}$, and $\{|e_3\rangle, |e_4\rangle\}$. In other words, we might be trying to distinguish one *subspace* from another, without separating vectors that lie in the same subspace. Such measurements (said to be **incomplete**) are indeed possible, and they can be less disruptive than the complete measurements.

Intuitively, an incomplete measurement has fewer outcomes and is hence less informative, but the state after such a measurement is usually less disturbed.

In general, instead of projecting on one dimensional subspaces spanned by vectors from an orthonormal basis, we can decompose our Hilbert space into mutually orthogonal subspaces of various dimensions and project onto them.

A full system of projectors satisfies two conditions: Conditions on *projectors*:

- **Orthogonality:** $P_k P_l = P_k \delta_{kl}$
- **Completeness:** $\sum_k P_k = \mathbf{1}$

For any decomposition of the identity into orthogonal projectors P_k (using the completeness condition), there exists a measurement that takes a quantum system in state $|\psi\rangle$, gives the output labelled k with probability $\langle \psi | P_k | \psi \rangle$, and leaves the system in the state $P_k |\psi\rangle$ (multiplied by the normalisation factor, i.e. divided by the length of $P_k |\psi\rangle$):

$$|\psi\rangle \mapsto \frac{P_k |\psi\rangle}{\sqrt{\langle \psi | P_k | \psi \rangle}}.$$

4.4 Example of an incomplete measurement

Take a three-dimensional Hilbert space \mathcal{H} with basis $\{|e_1\rangle, |e_2\rangle, |e_3\rangle\}$, and consider the two orthogonal projectors

$$\begin{aligned} P &= |e_1\rangle \langle e_1| + |e_2\rangle \langle e_2| \\ Q &= |e_3\rangle \langle e_3| \end{aligned}$$

These form the decomposition of the identity: $P + Q = \mathbf{1}$. Now suppose that a physical system is prepared in state $|\psi\rangle = \alpha_1 |e_1\rangle + \alpha_2 |e_2\rangle + \alpha_3 |e_3\rangle$. Ideally, we would like to perform a complete measurement that would resolve the state $|\psi\rangle$ into the three basis states, but suppose our experimental apparatus is not good enough, and lumps together $|e_1\rangle$ and $|e_2\rangle$. In other words, it can only differentiate between the two subspaces associated with projectors P and Q .

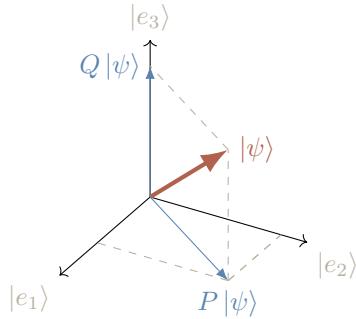
The apparatus, in this incomplete measurement, may find the system in the subspace associated with P . This happens with probability

$$\begin{aligned}\langle \psi | P | \psi \rangle &= \langle \psi | e_1 \rangle \langle e_1 | \psi \rangle + \langle \psi | e_2 \rangle \langle e_2 | \psi \rangle \\ &= |\alpha_1|^2 + |\alpha_2|^2,\end{aligned}$$

and the state right after the measurement is the normalised vector $P|\psi\rangle$, i.e.

$$\frac{\alpha_1|e_1\rangle + \alpha_2|e_2\rangle}{\sqrt{|\alpha_1|^2 + |\alpha_2|^2}}.$$

The measurement may also find the system in the subspace associated with Q with the probability $\langle \psi | Q | \psi \rangle = |\alpha_3|^2$, resulting in the post-measurement state $|e_3\rangle$.



4.5 Observables

An **observable** A is a measurable physical property which has a numerical value, for example, spin, position, momentum, or energy. The term “observable” also extends to any basic measurement in which each outcome has an associated numerical value. If λ_k is the numerical value associated to the outcome $|e_k\rangle$, then the observable A is represented by the operator

$$\begin{aligned}A &= \sum_k \lambda_k |e_k\rangle \langle e_k| \\ &= \sum_k \lambda_k P_k,\end{aligned}$$

where λ_k is now the eigenvalue corresponding to the eigenvector $|e_k\rangle$, or to the projector P_k .

We have already seen the following types of operators:

normal	$AA^\dagger = A^\dagger A$
unitary	$A^\dagger = A^{-1}$
Hermitian (or self-adjoint)	$A^\dagger = A$
positive semi-definite	$\langle v A v \rangle \geq 0$ for all $ v\rangle$

The **spectral theorem** says that an operator A is normal if and only if it is **unitarily diagonalisable**: there exists some *unitary* U and some *diagonal* D such that $A = U^\dagger D U$.

Note that unitary, Hermitian, and positive semi-definite operators are all, in particular, normal.

Since $(|a\rangle\langle b|)^\dagger = |b\rangle\langle a|$, the projectors $P_k = |e_k\rangle\langle e_k|$ are Hermitian, and thus normal, which means that A itself is a normal operator.

Conversely, given any normal operator A , we can associate a measurement defined by the eigenvectors of A , which form an orthonormal basis, and use the eigenvalues of A to label the outcomes of this measurement. If we choose the eigenvalues to be real numbers then A becomes a Hermitian operator. For example, the standard measurement on a single qubit is often called the **Z -measurement**, because the Pauli Z operator can be diagonalised in the standard basis and written as $Z = (+1)|0\rangle\langle 0| + (-1)|1\rangle\langle 1|$. The two outcomes, $|0\rangle$ and $|1\rangle$, are now labelled as $+1$ and -1 , respectively. Using the same association we also have the X - and the Y -measurements, defined by the Pauli X and Y operators, respectively.

The outcomes can be labelled by *any symbols of your choice* — it is the *decomposition* of the Hilbert space into *mutually orthogonal subspaces* that defines a measurement, not the labels.

This said, labelling outcomes with real numbers is very useful. Some textbooks describe observables in terms of Hermitian operators, claiming that the corresponding operators *have to* be Hermitian “because the outcomes are real numbers”. This is actually a bit backwards. As we say above, the labels can be arbitrary, but, since real number labels are often useful (as we’re about to justify), we tend to only work with Hermitian operators.

For example, the **expected value** $\langle A \rangle$ (also known as the **mean**), which is the average of the numerical values λ_k weighted by their probabilities, is a very useful quantity and can be easily expressed in terms of the operator A and the state of the system $|\psi\rangle$ as follows:

$$\begin{aligned}\langle A \rangle &= \sum_k \lambda_k \Pr(k) \\ &= \sum_k \lambda_k |\langle e_k | \psi \rangle|^2 \\ &= \sum_k \lambda_k \langle \psi | e_k \rangle \langle e_k | \psi \rangle \\ &= \langle \psi | \left(\sum_k \lambda_k |e_k\rangle\langle e_k| \right) |\psi\rangle \\ &= \langle \psi | A | \psi \rangle.\end{aligned}$$

It is important to note here that the notation $\langle A \rangle$ is slightly misleading, as it omits the dependence on the initial state $|\psi\rangle$. Some authors thus write $\langle A \rangle_{|\psi\rangle}$ instead, but many opt (as we do) for the more succinct notation.

To be clear, this is not a value we expect to see in one particular run of the experiment, but instead a statistical average. Imagine a huge number of quantum objects, all prepared in the state $|\psi\rangle$ and think about the observable A being measured on each of the objects. Statistically, we expect the average of our measurement results to be roughly $\langle A \rangle$. Note that when A is, in particular, a single projector $A = \lambda_k |e_k\rangle\langle e_k|$ then $\langle \psi | A | \psi \rangle$ is the probability of the outcome associated with A .

4.6 Compatible observables and the uncertainty relation

Now that we have explained how observables correspond to normal operators, we can try to understand what implications follow from the fact that matrix multiplication does *not* generally commute: $AB \neq BA$. We can start by trying to figure out when exactly two given operators A and B will or will not commute, ideally in terms of eigenvectors (since this will let us talk about outcomes and their numerical values, using the language we have just built up). An important definition is the following: if a basis $\{|e_1\rangle, \dots, |e_n\rangle\}$ is such that each $|e_k\rangle$ is an eigenvector of an operator A , then we call it an **eigenbasis of A** .

First of all, assume that A and B do commute, so that $AB = BA$, and let $|e\rangle$ be some eigenvector of A with eigenvalue λ . Then

$$\begin{aligned} AB|e\rangle &= BA|e\rangle \\ &= B\lambda|e\rangle \\ &= \lambda(B|e\rangle) \end{aligned}$$

which says that $B|e\rangle$ is also an eigenvector of A , with eigenvalue λ . If $\lambda \neq 0$, then this says that $B|e\rangle$ is proportional to $|e\rangle$, which is simply saying that $|e\rangle$ is also an eigenvector of B . This means that any eigenbasis of A is also an eigenbasis of B . Another way of saying this is that A and B are **simultaneously diagonalisable**: there exists a basis in which both A and B are diagonal, namely any common eigenbasis of the two.

Conversely, say that A and B have some common eigenbasis $\{|e_1\rangle, \dots, |e_n\rangle\}$, with $A|e_k\rangle = \alpha_k|e_k\rangle$ and $B|e_k\rangle = \beta_k|e_k\rangle$. To show that $AB = BA$, it suffices to show that $(AB)|\psi\rangle = (BA)|\psi\rangle$ for any state $|\psi\rangle$. But we can write any $|\psi\rangle$ in the common eigenbasis as $|\psi\rangle = \sum_k \lambda_k|e_k\rangle$ for some λ_k , and then

$$\begin{aligned} (AB)|\psi\rangle &= AB \sum_k \lambda_k|e_k\rangle \\ &= \sum_k \lambda_k AB|e_k\rangle \\ &= \sum_k \lambda_k A\beta_k|e_k\rangle \\ &= \sum_k \lambda_k \beta_k A|e_k\rangle \\ &= \sum_k \lambda_k \beta_k \alpha_k|e_k\rangle \end{aligned}$$

and α_k and β_k commute, since they are just complex numbers. This means that running the same calculation for $(BA)|\psi\rangle$ would give exactly the same result, and so $AB = BA$.

Two operators A and B commute if and only if there exists some common eigenbasis. In this case, we say that A and B are **compatible**; if A and B do not commute then we say that they are **incompatible**.

We have said that eigenvectors $|e_k\rangle$ of an operator A correspond to outcomes of the observable, where the eigenvalue λ_k is the associated numerical value. So if we have two compatible operators A and B , then we have a complete system of measurements for *both* observables at once, given by their common eigenbasis, say $\{|e_1\rangle, \dots, |e_k\rangle\}$. What does this mean in terms of measurements? Well, if we measure A on some system initially in state $|\psi\rangle$, then we know that the system will collapse into one of the states $|e_k\rangle$. But this is also an eigenvector for B , so measuring B won't affect the state at all, and similarly for a subsequent measurement of A .

If, however, A and B are incompatible operators, then things are very different. If we measure A , then B , and then A again, there is absolutely no guarantee that the two measurements of A will be the same. In other words, measuring B somehow makes the system "forget" the result of the first measurement of A . We see this in the lab if we measure *position* and *momentum* of a particle: taking the momentum measurement "spreads out" the position of the particle throughout space, meaning that a position measurement taken immediately prior will have no reason to be the same as a position measurement taken immediately afterwards.

Incompatible operators turn up all over the place, and actually turn out to be very interesting — sometimes it's good when things don't work too simply! One particularly

To make this argument fully formal, and to deal with the case where λ is degenerate, isn't too hard, but we don't want to get too involved with the necessary linear algebra here.

interesting question we can ask is the following: *can we quantify how far away from being compatible two incompatible operators are?* We can make this question more mathematically concrete by rephrasing it slightly, asking if we can find at least *some* states that are *close* to being common eigenstates.

Imagine preparing a huge number of systems into the same initial state $|\psi\rangle$, and then measuring A on half of them and B on the other half. Doing so we can obtain the expected values $\langle A \rangle$ and $\langle B \rangle$, and we can calculate (using classical statistics) the **standard deviation** of these variables, σ_A and σ_B , respectively. The standard deviation of a random variable is basically a measurement of “how close to the expected value are all the resulting values”. The smaller the standard deviation, the more “well defined” the measurement is. In particular, given any single operator A , we can always make the standard deviation exactly 0, by just preparing our system in an eigenstate of A . If A and B are compatible, then we can simultaneously make σ_A and σ_B exactly 0 as well, since we know that A and B have a common eigenbasis.

The really interesting, purely quantum, phenomena, however, comes when A and B are incompatible: we can prove that the standard deviations cannot both be made simultaneously arbitrarily small.

For example, if the random variable is normally distributed, then around 68% of the results will lie within one standard deviation from the expected value.

The **uncertainty principle** for operators A and B says that

$$\sigma_A \sigma_B \geq \left| \frac{1}{2i} \langle [A, B] \rangle \right|$$

where $[A, B] = AB - BA$ is the **commutator**.

This says that there does not exist *any* state for which $\sigma_A \sigma_B$ is less than some specific value, which is determined entirely by the operators A and B . Of course, if A and B are compatible, then $[A, B] = 0$, and so the uncertainty principle doesn’t tell us anything at all — it simply says that the product of two non-negative numbers is greater than or equal to 0, which is always the case!

You have maybe heard elsewhere of **Heisenberg’s uncertainty principle**, which is indeed a special case of this: one can show that the commutator of the (one-dimensional) position and momentum operators is exactly $i\hbar$ (where \hbar is again the very small number known as the **Planck constant**), whence $\sigma_x \sigma_p \geq \frac{\hbar}{2}$.

Quantization.

We said that \hbar is very small, and this is fundamental to the relationship between quantum and classic physics. Most of the things that we deal with in day-to-day life are on the macroscopic level, and are many, many orders of magnitude larger than the Planck constant. Indeed, if we wave our hands quite a lot, then we can say that “we see quantum effects only when dealing with things on the same order of magnitude as the Planck constant”. For example, a single photon of green light (roughly midway through the visible spectrum) has energy $\approx 3.5 \times 10^{-19}$ joules, whereas a mole of such photons (which is a “reasonable” number to encounter when talking about things that actually look green in day-to-day life) has energy $\approx 200 \times 10^3$ joules, so we would expect a single photon to exhibit quantum behaviour much more measurably than, for example, the light emitted from a green light bulb.

In a way which we shall not make precise, the fact that \hbar is strictly greater than zero (albeit very small) is what makes quantum physics inherently *discrete*, in contrast to classical physics which treats things like energy *continuously*. Quite wondrously, it is very often the case that taking a

limit $\hbar \rightarrow 0$ in some formula in quantum physics recovers the corresponding formula in classical physics — this is known as the **classical limit** or **correspondence principle**. This isn't unique to quantum physics: special relativity reduces to classical mechanics if we take all velocities to be much smaller than the speed of light; general relativity reduces to the classical theory of gravity if we take all gravitational fields to be weak enough; statistical mechanics reduces to thermodynamics when we take the number of particles to be large enough; and so on.

This idea, that classical systems can be recovered from quantum ones by taking $\hbar \rightarrow 0$, poses a question: *can we go in the other direction?* That is, given some classical theory that we know agrees with physical experiments, can we formulate some corresponding quantum version which we might hope to be correct on much smaller scales? Trying to answer this question has led to some incredibly deep (and very technical) mathematics known as **quantization theory**, with **geometric quantization** and **deformation quantization** being two key areas.

Before moving on, let us consider one more quantum phenomena that arises when we look at incompatible operators. Suppose that we have three operators, say A , B , and C , and we wish to let these act on our quantum system sequentially, but throwing away any results which are not a given outcome. That is, if we start (for simplicity) with some eigenstate $|a\rangle$ of A , then we want to know the probability of measuring some specific output $|c\rangle$. But we know how to calculate this!

First of all, we know the probability of measuring outcome $|c\rangle$ given that $|a\rangle$ first evolves into the intermediate state $|b_k\rangle$: this is the probability of $|a\rangle$ evolving under B into $|b_k\rangle$ multiplied by the probability of $|b_k\rangle$ evolving under C into $|c\rangle$, i.e.

$$\Pr(c|b_k) = |\langle c|b_k\rangle|^2 |\langle b_k|a\rangle|^2.$$

Then to obtain the probability of measuring outcome $|c\rangle$ we can just sum over all possible intermediate states:

$$\Pr(c) = \sum_k |\langle c|b_k\rangle|^2 |\langle b_k|a\rangle|^2.$$

But now, if we forget entirely about B then we could calculate $\Pr(c)$ in a different way: it is simply given by

$$\Pr(c) = |\langle c|a\rangle|^2.$$

Using the fact that $\sum_k |b_k\rangle\langle b_k| = \mathbf{1}$, we can rewrite this as

$$\Pr(c) = \left| \sum_k \langle c|b_k\rangle \langle b_k|a\rangle \right|^2$$

and this is *not generally equal* to the previous expression for $\Pr(c)$. In fact, you can show that these two expressions agree if and only if $[A, B] = 0$ or $[B, C] = 0$, i.e. if and only if either A and B or B and C are compatible.

We briefly discuss an explicit scenario of where three evolutions behave in such a paradoxical way later on in Chapter 6, when we introduce Bell's theorem, in what is sometimes known as the **quantum Venn diagram paradox**.

4.7 Quantum communication

Now is a good moment to introduce Alice and Bob (not their real names): our two protagonists who always need to communicate with each other, in scenarios of varying complexity and danger. These two play the major role in many communication

dramas, though they remain rather lacking in character development. In this episode of their story, Alice is sending quantum states, called **carriers**, to Bob, and Bob is trying his best to correctly identify them by choosing appropriate measurements.

Let us start with a simple observation: if the carriers are described by state vectors in a 2^n -dimensional Hilbert space, then they can encode at most n bits of information. For example, Alice can choose one of the 2^n states from a pre-agreed orthonormal basis $\{|e_k\rangle\}_{k=1,\dots,2^n}$, and Bob will be able to distinguish them reliably by choosing the same basis for his measurement.

But can Alice and Bob do better than that? Can Alice send *more* than n bits of information per carrier by encoding them in states $|s_1\rangle, \dots, |s_N\rangle$ where $N \geq 2^n$? Can Bob choose a clever measurement and reliably distinguish between all such states?

The answer is *no*.

This is just like the classical scenario: the space of binary strings of length n (which encode exactly n bits of information, by definition) is of dimension 2^n , since we describe any such string by picking between 0 and 1 for each digit, and we have n -many digits.

4.8 Basic quantum coding and decoding

Suppose Alice uniformly at random chooses one of the pre-agreed N signal states $|s_1\rangle, \dots, |s_N\rangle$ and sends it to Bob, who tries to identify the signal states by performing a measurement defined by the projectors P_1, \dots, P_N . Let P be a projector on the subspace spanned by the signal states $|s_1\rangle, \dots, |s_N\rangle$, i.e. $P|s_k\rangle = |s_k\rangle$ for all $k = 1, \dots, N$. The dimension d of this subspace is given by $d = \text{tr } P$. We shall assume, without any loss of generality, that Bob designed his measurement in such a way that, whenever he gets outcome P_k , he concludes that Alice sent state $|s_k\rangle$. His probability of successfully identifying which state Alice sent to him is given by

$$\Pr(\text{success}) = \frac{1}{N} \sum_k \langle s_k | P_k | s_k \rangle$$

which is the probability that signal state $|s_k\rangle$ is selected (here equal to $1/N$, since Alice chose between all N signal states with equal probability) times the probability that the selected signal state is correctly identified by Bob (which is $\langle s_k | P_k | s_k \rangle$), and we sum over all possible signal states.

Let us use this as a chance to practice some of the trace identities. In particular, it is often convenient to write expressions such as $\langle \psi | A | \psi \rangle$ in terms of the trace: for any vector $|\psi\rangle$ and operator A we have

$$\begin{aligned} \langle \psi | A | \psi \rangle &= \text{tr}(A|\psi\rangle\langle\psi|) \\ &= \text{tr}(|\psi\rangle\langle\psi|A). \end{aligned}$$

In our case,

$$\begin{aligned} \Pr(\text{success}) &= \frac{1}{N} \sum_k \langle s_k | P_k | s_k \rangle \\ &= \frac{1}{N} \sum_k \langle s_k | P P_k P | s_k \rangle \\ &= \frac{1}{N} \sum_k \text{tr}(P P_k P | s_k \rangle \langle s_k |) \end{aligned}$$

where we have also used that $P|s_k\rangle = |s_k\rangle$.

If B is a positive semi-definite operator, and P is a projector, then

$$\mathrm{tr} \, BP \leqslant \mathrm{tr} \, B.$$

To prove this, consider the projector $Q = \mathbf{1} - P$, and note that

$$\begin{aligned}\mathrm{tr} \, B &= \mathrm{tr} \, B(P + Q) \\ &= \mathrm{tr} \, BP + \mathrm{tr} \, BQ\end{aligned}$$

and that $\mathrm{tr} \, BQ$ is non-negative.

We can use this inequality to bound the expression above:

$$\begin{aligned}\sum_k \frac{1}{N} \langle s_k | P_k | s_k \rangle &= \frac{1}{N} \sum_k \mathrm{tr}(PP_kP|s_k\rangle\langle s_k|) \\ &\leqslant \frac{1}{N} \sum_k \mathrm{tr}(PP_kP) \\ &= \frac{1}{N} \mathrm{tr} \left(P \left(\sum_k P_k \right) P \right) \\ &= \frac{1}{N} \mathrm{tr}(P^3) \\ &= \frac{1}{N} \mathrm{tr}(P) \\ &= \frac{d}{N}.\end{aligned}$$

So if Alice encodes N equally likely messages as states in a quantum system that, mathematically speaking, lives in the Hilbert space of dimension d , and if Bob decodes by performing a measurement and inferring the message from the result, then Bob's probability of success is bounded above by $\frac{d}{N}$. *If the number N of possible signals exceeds the dimension d , then Bob will not be able to reliably distinguish between the signals by any measurement.* In particular:

With this setup, one qubit can store *at most* one bit of information that can *reliably* be read by a measurement.

4.9 Distinguishing non-orthogonal states

We have already mentioned (Section 4.3) that non-orthogonal states cannot be reliably distinguished, and now we can make this statement more precise. Suppose Alice sends Bob a message by choosing one of the two *non-orthogonal* states $|s_1\rangle$ and $|s_2\rangle$, where both are equally likely to be chosen. What is the probability that Bob will decode the message correctly, and what is the best (i.e. the one that maximises this probability) choice of measurement?

There is something called **super-dense coding**, where one qubit can actually store two classical bits, but this relies on Alice and Bob both having access to a shared entangled state right from the very start of the experiment. We shall eventually study this in Exercise 5.14.9.

As a general rule, before you embark on any calculations, check for symmetries, special cases, and anything that may help you to visualise the problem and make intelligent guesses about the solution. One of the most powerful research tools is a good guess! In fact, this is what real research is about: educated guesses that guide your calculations. In this particular case you can use symmetry arguments to guess the optimal measurement — see Figure 4.2. Once you have guessed the answer, you might as well do the calculations.

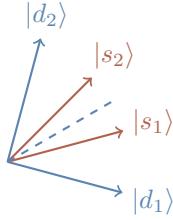


Figure 4.2: The optimal measurement to distinguish between the two equally likely non-orthogonal signal states $|s_1\rangle$ and $|s_2\rangle$ is described by the two orthogonal vectors $|d_1\rangle$ and $|d_2\rangle$ placed symmetrically around the signal states.

Thinking about what we have already seen, we should expect that how well we can correctly distinguish between $|s_1\rangle$ and $|s_2\rangle$ is directly proportional to “how close” they are to being orthogonal — if they are orthogonal, then we can distinguish perfectly; if they are identical (i.e. collinear), then we cannot distinguish between them at all. Hopefully, then, our final answer will depend on the angle between $|s_1\rangle$ and $|s_2\rangle$.

So suppose Bob’s measurement is described by projectors P_1 and P_2 , chosen such that “ P_1 implies $|s_1\rangle$, and P_2 implies $|s_2\rangle$ ”. Then

$$\begin{aligned}\Pr(\text{success}) &= \frac{1}{2} (\langle s_1 | P_1 | s_1 \rangle + \langle s_2 | P_2 | s_2 \rangle) \\ &= \frac{1}{2} (\text{tr } P_1 |s_1\rangle\langle s_1| + \text{tr } P_2 |s_2\rangle\langle s_2|) \\ &= \frac{1}{2} (\text{tr } P_1 |s_1\rangle\langle s_1| + \text{tr } (\mathbf{1} - P_1) |s_2\rangle\langle s_2|) \\ &= \frac{1}{2} (1 + \text{tr } P_1 (|s_1\rangle\langle s_1| - |s_2\rangle\langle s_2|)).\end{aligned}$$

Let us look at the operator $D = |s_1\rangle\langle s_1| - |s_2\rangle\langle s_2|$ that appears in the last expression. This operator acts on the subspace spanned by $|s_1\rangle$ and $|s_2\rangle$; it is Hermitian; the sum of its two (real) eigenvalues is zero (whence $\text{tr } D = \langle s_1 | s_1 \rangle - \langle s_2 | s_2 \rangle = 0$). Let us write D as $\lambda(|d_+\rangle\langle d_+| - |d_-\rangle\langle d_-|)$, where $|d_\pm\rangle$ are the two orthonormal eigenstates of D , and $\pm\lambda$ are the corresponding eigenvalues.

Now we write

$$\begin{aligned}\Pr(\text{success}) &= \frac{1}{2} (1 + \lambda \text{tr } P_1 (|d_+\rangle\langle d_+| - |d_-\rangle\langle d_-|)) \\ &\leq \frac{1}{2} (1 + \lambda \langle d_+ | P_1 | d_+ \rangle)\end{aligned}$$

where we have dropped the non-negative term $\text{tr } P_1 |d_-\rangle\langle d_-|$. In fact, it is easy to see that we will maximise the expression above by choosing $P_1 = |d_+\rangle\langle d_+|$ and $P_2 = |d_-\rangle\langle d_-|$. The probability of success is then bounded by $\frac{1}{2}(1 + \lambda)$. All we have to do now is to find the positive eigenvalue λ for the operator D .

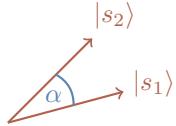
We can do this, of course, by solving the characteristic equation for a matrix representation of D , but, since we are practising using the trace identities, we can also notice that $\text{tr } D^2 = 2\lambda^2$, and then evaluate the trace of D^2 . We use the trace identities and obtain

$$\begin{aligned}\text{tr } D^2 &= \text{tr } (|s_1\rangle\langle s_1| - |s_2\rangle\langle s_2|)(|s_1\rangle\langle s_1| - |s_2\rangle\langle s_2|) \\ &= 2 - 2|\langle s_1 | s_2 \rangle|^2\end{aligned}$$

which gives $\lambda = \sqrt{1 - |\langle s_1 | s_2 \rangle|^2}$. Bringing it all together we have the final expression:

$$\Pr(\text{success}) \leq \frac{1}{2} \left(1 + \sqrt{1 - |\langle s_1 | s_2 \rangle|^2} \right).$$

We can parametrise $|\langle s_1 | s_2 \rangle| = \cos \alpha$, where α is then the angle between $|s_1\rangle$ and $|s_2\rangle$.



This allows us to express our findings in a clearer way: given two equally likely states, $|s_1\rangle$ and $|s_2\rangle$, such that $|\langle s_1 | s_2 \rangle| = \cos \alpha$, the probability of correctly identifying the state by a projective measurement is bounded by

$$\Pr(\text{success}) \leq \frac{1}{2}(1 + \sin \alpha),$$

Here we use that $\cos^2 \alpha + \sin^2 \alpha = 1$ for any α .

and the optimal measurement that achieves this bound is determined by the eigenvectors of $D = |s_1\rangle\langle s_1| - |s_2\rangle\langle s_2|$ (try to visualise these eigenvectors).

It makes sense, right? If we try just guessing the state, without any measurement, then we expect $\Pr(\text{success}) = \frac{1}{2}$. This is our lower bound, and in any attempt to distinguish the two states we should do better than that. If the two signal states are very close to each other, then $\sin \alpha$ is small and we are slightly better off than guessing. As we increase α , the two states become more distinguishable, and, as we can see from the formula, when the two states become orthogonal they also become completely distinguishable.

We will return to this same problem later on, in Section 12.8, where we will use a different, less ad-hoc, approach, working in the more general setting of so-called **density operators**.

4.10 Wiesner's quantum money

This section is not yet finished.

4.11 Quantum theory, formally

Even though multiplying and adding probability amplitudes is essentially all there is to quantum theory, we hardly ever multiply and add amplitudes in a pedestrian way. Instead, as we have seen, we neatly tabulate the amplitudes into vectors and matrices and let the matrix multiplication take care of multiplication and addition of amplitudes corresponding to different alternatives. Thus vectors and matrices appear naturally as our bookkeeping tools: we use vectors to describe quantum states, and matrices (operators) to describe quantum evolutions and measurements. This leads to a convenient mathematical setting for quantum theory: a complex vector space with an inner product (which is exactly a Hilbert space, since we only work in finite dimension). It turns out, somewhat miraculously, that this pure mathematical construct is exactly what we need to formalise quantum theory. It gives us a precise language which is appropriate for making empirically testable predictions. At a very instrumental level, quantum theory is a set of rules designed to answer questions such as “given a specific preparation and a subsequent evolution, how can we compute probabilities for the outcomes of such-and-such measurement”. Here is how we represent preparations, evolutions and measurements in mathematical terms, and how we get probabilities.

Note that we have already said much of the below, but we are summarising it again now in a more precise way, formally defining the mathematical framework of quantum theory that we use.

We also need to point out that a vital part of the formalism of quantum theory is missing from the following description, namely the idea of **tensor products**. To talk about this, we need to introduce the notion of **entanglement**, and this will be the subject of the next chapter.

Axiomatic quantum theory.

It is a very reasonable question to ask *why* this formalism (Hilbert spaces, unitary operators, the Born rule) is “the good one”. One answer is that “it just works” — the calculations that we do in this framework give us answers which are in agreement with the results of physical experiments — but this can be rather unsatisfying as an answer.

Quite beautifully, it turns out that if we start from just *five* axioms, then we can prove that our choice of formalism is actually the only one that makes sense. This is the result of L. Hardy’s “Quantum Theory From Five Reasonable Axioms”, arXiv:[quant-ph/0101012](https://arxiv.org/abs/quant-ph/0101012). We start by saying that a quantum system should be characterised by two integers: the number of degrees of freedom K , and the dimension N . The former is (roughly) the minimum number of real numbers needed to specify any state; the latter is the maximum number of states that can be distinguished from one another in one single measurement. The five axioms are then as follows.

1. *Probabilities.* Relative frequencies of observed outcomes from measuring an ensemble of n systems tend to a well defined value, called the **probability**, when n tends to infinity.
2. *Simplicity.* The integer K is a function of N , and takes the minimum possible value consistent with these axioms for each N .
3. *Subspaces.* If a system is such that its states all lie within an M -dimensional subspace (for some $M < N$), then it behaves exactly like a system of dimension M .
4. *Composite systems.* Composite systems behave multiplicatively, i.e. if a system is a composite of two subsystems A and B , then $N = N_A N_B$ and $K = K_A K_B$.
5. *Continuity.* Given any two **pure states** (all of the states that we have been discussing so far are pure states, but we define what this means in Section 8.1.) of a system, there exists a continuous reversible transformation of the system that sends one to the other.

What is particularly nice, as a bonus result, is that if we make one tiny change to these axioms — just dropping the word “continuous” from the fifth axiom — then the result is exactly *classical* probability theory.

Quantum states

With any isolated quantum system which can be prepared in n perfectly distinguishable states, we can associate a Hilbert space \mathcal{H} of dimension n such that each vector $|v\rangle \in \mathcal{H}$ of unit length ($\langle v|v\rangle = 1$) represents a quantum state of the system. The overall phase of the vector has no physical significance: $|v\rangle$ and $e^{i\varphi}|v\rangle$, for any real φ , describe the same state. The inner product $\langle u|v\rangle$ is the probability amplitude that a quantum system prepared in state $|v\rangle$ will be found in state $|u\rangle$. States corresponding to orthogonal vectors, $\langle u|v\rangle = 0$, are perfectly distinguishable, since the system prepared in state $|v\rangle$ will never be found in state $|u\rangle$, and vice versa. In particular, states forming orthonormal bases are always perfectly distinguishable from each other.

Quantum evolutions

Any physically admissible evolution of an isolated quantum system is represented by a unitary operator.

Unitary operators describing evolutions of quantum systems are usually derived from the **Schrödinger equation**

$$\frac{d}{dt}|\psi(t)\rangle = -\frac{i}{\hbar}\hat{H}|\psi(t)\rangle$$

where \hat{H} is a Hermitian operator called the Hamiltonian.

This equation contains a complete specification of all interactions both within the system and between the system and the external potentials. For time-independent Hamiltonians, the formal solution of the Schrödinger equation reads

$$|\psi(t)\rangle = U(t)|\psi(0)\rangle$$

$$\text{where } U(t) = e^{-\frac{i}{\hbar}\hat{H}t}.$$

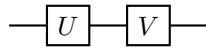
Any unitary matrix can be represented as the exponential of some Hermitian matrix \hat{H} and some real coefficient t :

$$\begin{aligned} e^{-it\hat{H}} &= \mathbf{1} - it\hat{H} + \frac{(-it)^2}{2}\hat{H}^2 + \frac{(-it)^3}{2 \cdot 3}\hat{H}^3 + \dots \\ &= \sum_{n=0}^{\infty} \frac{(-it)^n}{n!}\hat{H}^n. \end{aligned}$$

The state vector changes smoothly: for $t = 0$ the time evolution operator is merely the unit operator $\mathbf{1}$, and when t is very small $U(t) \approx \mathbf{1} - it\hat{H}$ is close to the unit operator, differing from it by something of order t .

Quantum circuits

In this course we will hardly refer to the Schrödinger equation. Instead we will assume that our clever colleagues — experimental physicists — are able to implement certain unitary operations, and we will use these unitaries, like lego blocks, to construct other, more complex, unitaries. We refer to pre-selected elementary quantum operations as **quantum logic gates** and we often draw diagrams, called **quantum circuits**, to illustrate how they act on qubits. For example, two unitaries, U followed by V , acting on a single qubit are represented as



This diagram should be read from left to right, and the horizontal line represents a qubit that is inertly carried from one quantum operation to another (maybe through space, down a physical wire, but maybe through some other physical implementation — we don't particularly mind!)

Measurements

A complete measurement in quantum theory is determined by the choice of an orthonormal basis $\{|e_1\rangle, \dots, |e_n\rangle\}$ in \mathcal{H} , and every such basis (in principle) represents a possible measurement. Given a quantum system in state $|\psi\rangle$ such that

$$|\psi\rangle = \sum_i |e_i\rangle\langle e_i| \psi \rangle,$$

the measurement in the basis $\{|e_1\rangle, \dots, |e_n\rangle\}$ gives the outcome labelled by e_k with probability $|\langle e_k|\psi\rangle|^2$, and leaves the system in state $|e_k\rangle$ after measurement. This is consistent with our interpretation of the inner product $\langle e_k|\psi\rangle$ as the probability amplitude that a quantum system prepared in state $|\psi\rangle$ will be found in state $|e_k\rangle$. State vectors forming orthonormal bases are perfectly distinguishable from each other

We briefly discussed this equation in Section 3.6.

$(\langle e_i | e_j \rangle = \delta_{ij})$, so there is no ambiguity about the outcome. A complete measurement is the best we can do in terms of resolving state vectors in the basis states.

In general, for any decomposition of the identity $\sum_k P_k = \mathbf{1}$ into orthogonal projectors P_k (i.e. $P_k P_l = P_k \delta_{kl}$), there exists a measurement that takes a quantum system in state $|\psi\rangle$, outputs label k with probability $\langle\psi|P_k|\psi\rangle$, and leaves the system in the state $P_k|\psi\rangle$ (multiplied by the normalisation factor i.e. divided by the length of $P_k|\psi\rangle$):

$$|\psi\rangle \mapsto \frac{P_k|\psi\rangle}{\sqrt{\langle\psi|P_k|\psi\rangle}}.$$

The projector formalism covers both complete and incomplete measurements. The complete measurements are exactly those defined by rank-one projectors $P_k = |e_k\rangle\langle e_k|$, projecting on vectors from some orthonormal basis $\{|e_k\rangle\}$.

4.12 Remarks and exercises

4.12.1 Projector?

Consider two unit vectors $|a\rangle$ and $|b\rangle$. Is the operator $|a\rangle\langle a| + |b\rangle\langle b|$ a projector?

4.12.2 Knowing the unknown

Suppose you are given a *single* qubit in some entirely unknown quantum state $|\psi\rangle = \alpha|0\rangle + \beta|1\rangle$.

1. Can you determine $|\psi\rangle$, using as many measurements as you want?
2. Say you measure the qubit in the standard basis, and register outcome $|0\rangle$. What does this tell you about the pre-measurement state $|\psi\rangle$?
3. How many real parameters do you need to determine $|\psi\rangle$? Would you be able to reconstruct $|\psi\rangle$ from $\langle\psi|X|\psi\rangle$, $\langle\psi|Y|\psi\rangle$, and $\langle\psi|Z|\psi\rangle$?
4. You are given zillions of qubits, all prepared in the same quantum state $|\psi\rangle$. How would you determine $|\psi\rangle$?

Hint: it may help you to visualise $|\psi\rangle$ as a Bloch vector.

4.12.3 Measurement and idempotents

The Z measurement is defined by the projectors

$$P_0 = \frac{1}{2}(\mathbf{1} + Z),$$

$$P_1 = \frac{1}{2}(\mathbf{1} - Z).$$

Let's generalise this.

Consider the measurement associated to any Hermitian operator S that satisfies $S^2 = \mathbf{1}$. Show that the two outcomes ± 1 correspond to the projectors $\frac{1}{2}(\mathbf{1} \pm S)$.

4.12.4 Unitary transformations of measurements

In our quantum circuits, unless specified otherwise, all measurements are assumed to be performed in the standard basis. This is because any measurement can be reduced to the standard measurement by performing some prior unitary transformation.

1. Show that any two orthonormal bases $\{|e_1\rangle, \dots, |e_n\rangle\}$ and $\{|d_1\rangle, \dots, |d_n\rangle\}$ are always related by some unitary U .

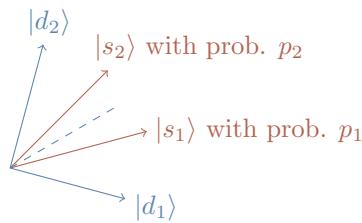
Hint: it suffices to show that $\sum_k |d_k\rangle\langle e_k|$ is unitary — why?

2. Suppose that the projectors P_k define the standard measurement. Show that, for any unitary U , the projectors UP_kU^\dagger also define a measurement.

$$|\psi\rangle \xrightarrow{UP_kU^\dagger} \quad \equiv \quad |\psi\rangle \xrightarrow{U} \xrightarrow{P_k}$$

4.12.5 Optimal measurement

The optimal measurement to distinguish between the two equally likely non-orthogonal signal states, $|s_1\rangle$ and $|s_2\rangle$, is described by the two orthogonal vectors $|d_1\rangle$ and $|d_2\rangle$, placed symmetrically around the signal states, as we saw in Section 4.9. But suppose the states are *not* equally likely: say $|s_1\rangle$ is chosen with probability p_1 and $|s_2\rangle$ with probability p_2 . How would you modify the measurement to maximise the probability of success in this case?



4.12.6 Alice knows what Bob did

Alice prepares a qubit in any state of her choosing and gives it to Bob, who secretly measures either σ_x or σ_y . The outcome of the measurement is seen only by Bob. Alice has no clue which measurement was chosen by Bob, but right after his measurement she gets her qubit back and she can measure it as well. Some time later, Bob tells Alice which of the two measurements was chosen, i.e. whether he measured σ_x or σ_y . Alice then tells him the outcome he obtained in his measurement. Bob is surprised, since the two measurements have mutually unbiased bases, and yet Alice always gets it right, no matter how many times they repeat the experiment. How does she do it?

*This is a simplified version of a beautiful quantum puzzle proposed in 1987 by Lev Vaidman, Yakir Aharonov, and David Z. Albert in a paper with the somewhat provocative title “How to ascertain the values of σ_x , σ_y , and σ_z of a spin- $\frac{1}{2}$ particle”. For the original, see Phys. Rev. Lett. **58** (1987), p. 1385.*

4.12.7 The Zeno effect

This section is not yet finished.

5 Entanglement

*About the fundamental tool of quantum computing: **entanglement**, via the formalism of **tensor products**, which was the missing ingredient from our previous formalism of quantum theory. Also about various **controlled gates**, including the always useful **controlled-NOT**.*

We now know everything we need to know about a single qubit and its quantum behaviour. But if we want to understand quantum computation — a complicated quantum interference of many interacting qubits — then we will need few more mathematical tools. Stepping up from one qubit to two or more is a bigger leap than you might expect. Already, with just two qubits, we will encounter the remarkable phenomenon of **quantum entanglement** and have a chance to discuss some of the most puzzling features of quantum theory that took people decades to understand.

5.1 A very brief history

The notion of **quantum entanglement** was the subject of many early debates that focused on the meaning of quantum theory. Back in the 1930s, Albert Einstein, Niels Bohr, Werner Heisenberg, and Erwin Schrödinger (to mention just the usual suspects) were trying hard to understand its conceptual consequences. Einstein, the most sceptical of them all, claimed that it was pointing toward the fatal flaw in quantum theory, and referred to it as “spooky action at a distance” (“*spukhafte Fernwirkung*”). In contrast, Schrödinger was much more prepared to accept quantum theory exactly as it was formulated, along with all its predictions, no matter how weird they might be. In his 1935 paper, which introduced quantum entanglement, he wrote “I would not call it *one* but rather *the* characteristic trait of quantum mechanics, the one that enforces its entire departure from classical lines of thought”.

Today we still talk a lot about quantum entanglement, but more often it is viewed as a physical resource which enables us to communicate with perfect security, build very precise atomic clocks, and even teleport small quantum objects! But what exactly is quantum entanglement?

5.2 From one qubit to two

In classical physics, the transition from a single object to a composite system of many objects is trivial: in order to describe the state of, say, 42 objects at any given moment of time, it is sufficient to describe the state of each of the objects separately. Indeed, the classical state of 42 point-like particles is described by specifying the position and the momentum of each particle.

In the *classical* world, “the whole is *exactly* the sum of its parts”; in the *quantum* world, Aristotle had it right when he said “the whole is *greater than* the sum of its parts”.

Consider, for example, a pair of qubits. Suppose that each one is described by a state vector: the first one by $|a\rangle$, and the second one by $|b\rangle$. One might therefore think that the most general state of the two qubits should be represented by a pair of state vectors, $|a\rangle|b\rangle$, with one for each qubit. Indeed, such a state is certainly possible, but there are other states that *cannot* be expressed in this form. In order to write down the most general state of two qubits we first focus on the basis states.

For a single qubit we have been using the standard basis $\{|0\rangle, |1\rangle\}$. For two qubits we may choose the following as our standard basis states:

For our purposes! Of course, there is a *lot* that we could still ask, but we leave these questions to quantum physicists, or scientists working in a lab.

E. Schrödinger, “Discussion of probability relations between separated system”. *Mathematical Proceedings of the Cambridge Philosophical Society* **31** (1935), pp. 555–563.

$$\begin{aligned} |00\rangle &\equiv |0\rangle|0\rangle & |01\rangle &\equiv |0\rangle|1\rangle \\ |10\rangle &\equiv |1\rangle|0\rangle & |11\rangle &\equiv |1\rangle|1\rangle. \end{aligned}$$

Within each ket, the first symbol refers to the first qubit, and the second to the second, and we have tacitly assumed that we can distinguish the two qubits by their location, or some other means.

Now, the most general state of the two qubits (a **bipartite** state) is a normalised linear combination of these four basis states, i.e. a vector of the form

$$|\psi\rangle = c_{00}|00\rangle + c_{01}|01\rangle + c_{10}|10\rangle + c_{11}|11\rangle.$$

Physical interpretation aside, let us count how many real parameters are needed to specify this state. Six, right? We have four complex numbers (the c_{ij}), which gives *eight* real parameters; we then restrict by the normalisation condition, along with the fact that states differing only by a global phase factor are equivalent, which leaves us with *six* real parameters. Now, by the same line of argument, we need only *two* real parameters to specify the state of a single qubit, and hence need *four* real parameters to specify any state of two qubits of the form $|a\rangle|b\rangle$.

But four is less than six! So it *cannot* be the case that every state of two qubits can be expressed as a pair of states $|a\rangle|b\rangle$, simply for “dimension reasons”.

For example, compare the two states of two qubits,

$$\frac{1}{\sqrt{2}}|00\rangle + \frac{1}{\sqrt{2}}|01\rangle \quad \text{and} \quad \frac{1}{\sqrt{2}}|00\rangle + \frac{1}{\sqrt{2}}|11\rangle.$$

The first one is **separable**, i.e. we can view it as a pair of state vectors where each one pertains to one of the two qubits:

$$\frac{1}{\sqrt{2}}|00\rangle + \frac{1}{\sqrt{2}}|01\rangle = \underbrace{\frac{1}{\sqrt{2}}}_{\text{qubit 1}} \underbrace{|0\rangle}_{\text{qubit 2}} \underbrace{(|0\rangle + |1\rangle)}_{\text{qubit 2}},$$

The second state, however, does *not* admit such a decomposition: there do *not* exist *any* ψ_1, ψ_2 such that

$$\frac{1}{\sqrt{2}}|00\rangle + \frac{1}{\sqrt{2}}|11\rangle = |\psi_1\rangle|\psi_2\rangle$$

and so we say that it is an **entangled** state.

Any bipartite state that cannot be viewed as a pair of two states pertaining to the constituent subsystems is said to be **entangled**.

We'll give another, equivalent but more mathematical (and notational), definition of entanglement once we understand how tensor products work.

5.3 Quantum theory, formally (continued)

In Section 4.11, we said that we were missing a key part in our formalism of quantum theory — now we can finally fill in this hole. Our mathematical formalism of choice behind the quantum theory of composite systems is based on the **tensor product** of Hilbert spaces.

Tensor products

Let the states of some system \mathcal{A} be described by vectors in an n -dimensional Hilbert space $\mathcal{H}_\mathcal{A}$, and the states of some system \mathcal{B} by vectors in an m -dimensional Hilbert space $\mathcal{H}_\mathcal{B}$. The combined system of \mathcal{A} and \mathcal{B} is then described by vectors in the nm -dimensional **tensor product space** $\mathcal{H}_\mathcal{A} \otimes \mathcal{H}_\mathcal{B}$. Given bases $\{|a_1\rangle, \dots, |a_n\rangle\}$ of $\mathcal{H}_\mathcal{A}$ and $\{|b_1\rangle, \dots, |b_m\rangle\}$ of $\mathcal{H}_\mathcal{B}$, we form a basis of the tensor product by taking the ordered pairs $|a_i\rangle \otimes |b_j\rangle$, for $i = 1, \dots, n$ and $j = 1, \dots, m$. For brevity, we sometimes write

$|a_i\rangle \otimes |b_j\rangle$ as $|a_i\rangle|b_j\rangle$, or simply $|a_i b_j\rangle$. The tensor product space $\mathcal{H}_A \otimes \mathcal{H}_B$ then consists of all linear combination of such tensor product basis vectors:

$$|\psi\rangle = \sum_{i,j} c_{ij} |a_i\rangle \otimes |b_j\rangle. \quad (\ddagger)$$

The tensor product operation \otimes is distributive:

$$\begin{aligned} |a\rangle \otimes (\beta_1|b_1\rangle + \beta_2|b_2\rangle) &= \beta_1|a\rangle \otimes |b_1\rangle + \beta_2|a\rangle \otimes |b_2\rangle \\ (\alpha_1|a_1\rangle + \alpha_2|a_2\rangle) \otimes |b\rangle &= \alpha_1|a_1\rangle \otimes |b\rangle + \alpha_2|a_2\rangle \otimes |b\rangle. \end{aligned}$$

The tensor product of Hilbert spaces is again a Hilbert space: the inner products on \mathcal{H}_A and \mathcal{H}_B give a natural inner product on $\mathcal{H}_A \otimes \mathcal{H}_B$, defined for any two product vectors by

$$(\langle a' | \otimes \langle b' |) (|a\rangle \otimes |b\rangle) = \langle a'|a\rangle \langle b'|b\rangle$$

and extended by linearity to sums of tensor products of vectors, and, by associativity, to any number of subsystems. Note that the bra corresponding to the tensor product state $|a\rangle \otimes |b\rangle$ is written as $(|a\rangle \otimes |b\rangle)^\dagger = \langle a| \otimes \langle b|$, where the order of the factors on either side of \otimes does *not* change when the dagger operation is applied.

Some joint states of \mathcal{A} and \mathcal{B} can be expressed as a single tensor product, say $|\psi\rangle = |a\rangle \otimes |b\rangle$, meaning that the subsystem \mathcal{A} is in state $|a\rangle$, and the subsystem \mathcal{B} in state $|b\rangle$. If we expand $|a\rangle = \sum_i \alpha_i |a_i\rangle$ and $|b\rangle = \sum_j \beta_j |b_j\rangle$, then $|\psi\rangle = \sum_{i,j} \alpha_i \beta_j |a_i\rangle \otimes |b_j\rangle$ and we see that, for all such states, the coefficients c_{ij} in Equation (\ddagger) are of a rather special form:

$$c_{ij} = \alpha_i \beta_j.$$

We call such states **separable** (or **product states**). States that are not separable are said to be **entangled**.

A useful fact about tensor products is that $\lambda a \otimes b = a \otimes \lambda b$ (where a and b are vectors, and λ is a scalar). This means that we don't need to worry about where exactly we put λ , and can write something like $\lambda(a \otimes b)$.

We will also need the concept of the tensor product of two operators. If A is an operator on \mathcal{H}_A and B an operator on \mathcal{H}_B , then the tensor product operator $A \otimes B$ is an operator on $\mathcal{H}_A \otimes \mathcal{H}_B$ defined by its action on product vectors via

$$(A \otimes B)(|a\rangle \otimes |b\rangle) = (A|a\rangle) \otimes (B|b\rangle)$$

and with its action on all other vectors determined by linearity:

$$A \otimes B \left(\sum_{i,j} c_{ij} |a_i\rangle \otimes |b_j\rangle \right) = \sum_{i,j} c_{ij} A|a_i\rangle \otimes B|b_j\rangle.$$

The universal property of the tensor product.

We have described the tensor product in terms of how it acts on bases, and then extended everything by linearity, distributivity, and associativity. But there are other, more abstract approaches to defining the tensor product.

For example, given two vector spaces V and W , we can construct their tensor product $V \otimes W$ as a *quotient* of the cartesian product $V \times W$ (whose elements are simply pairs (v, w) of vectors in V and vectors in W) by the

If the bases $\{|a_i\rangle\}$ and $\{|b_j\rangle\}$ are orthonormal then so too is the tensor product basis $\{|a_i\rangle \otimes |b_j\rangle\}$.

subspace spanned by the relations that we want the tensor product to satisfy:

$$\begin{aligned} (v_1 + v_2, w) - (v_1, w) - (v_2, w), \\ (v, w_1 + w_2) - (v, w_1) - (v, w_2), \\ (\lambda v, w) - \lambda(v, w), \\ (v, \lambda w) - \lambda(v, w). \end{aligned}$$

But really this is hinting at the so-called **universal property** that defines the tensor product without giving a choice of explicit construction: the tensor product of V and W is defined to be any vector space A along with a bilinear map $\otimes: V \times W \rightarrow A$ such that, for any other vector space Z along with a bilinear map $f: V \times W \rightarrow Z$, there exists a unique linear map $\tilde{f}: A \rightarrow Z$ such that $f = \tilde{f} \circ \otimes$. In the language of category theory, the tensor product is the **initial object** amongst vector spaces endowed with a bilinear map from $V \times W$; any other vector space Z with a bilinear map $V \times W \rightarrow Z$ factors through the tensor product.

One specific reason to care about giving a definition in terms of universal property is that this guarantees (by some **abstract nonsense**) that the resulting object will be unique (“up to unique isomorphism”) whenever it exists, so you don’t need to worry about proving this separately.

Tensor products are much more general than just for vector spaces: they can be **defined for modules** (which are like vector spaces over an arbitrary commutative ring, instead of over a field), and abelian groups are, it turns out, exactly “modules over \mathbb{Z} ”, so they also have a notion of tensor product. Going a bit deeper, we can define tensor products for **complexes** of modules and **sheaves** of modules, and these constructions are absolutely fundamental to modern algebraic geometry.

Going even deeper still (and now far beyond the purview of this book), tensor products are generalised by the notion of **monoidal categories**.

As a final note, the universal property of the tensor product can be used to prove that we do not need to impose the postulate “the Hilbert space of a composite system is the tensor product of the Hilbert spaces of its components”, but that this actually follows “for free” from the **state** and the **measurement postulates**. This is shown in Carcassi, Maccone, and Aidala’s “The four postulates of quantum mechanics are three”, arXiv:[2003.11007](#).

5.4 More qubits, and binary representations

Let’s see how this formalism works for qubits. The n -fold tensor product of vectors from the standard basis $\{|0\rangle, |1\rangle\}$ represent binary strings of length n . For example, for $n = 3$,

$$\begin{aligned} |0\rangle \otimes |1\rangle \otimes |1\rangle &\equiv |011\rangle \\ |1\rangle \otimes |1\rangle \otimes |1\rangle &\equiv |111\rangle. \end{aligned}$$

A **classical register** (that is, a collection of bits) composed of three bits can store *only one* of these two binary strings at any time; a **quantum register** composed of three qubits can store *both of them* in a superposition.

Indeed, if we start with the state $|011\rangle$ and apply the Hadamard gate to the first qubit (which is the same as applying $H \otimes \mathbf{1} \otimes \mathbf{1}$), then, given that linear combinations distribute over tensor products, we obtain

$$\begin{aligned} |011\rangle &\xrightarrow{H \otimes \mathbf{1} \otimes \mathbf{1}} \frac{1}{\sqrt{2}}(|0\rangle + |1\rangle) \otimes |1\rangle \otimes |1\rangle \\ &= \frac{1}{\sqrt{2}}(|011\rangle + |111\rangle). \end{aligned}$$

In fact, we can even prepare this register in a superposition of all eight possible binary strings of length 3 at once: if we apply the tensor product operation $H \otimes H \otimes H$ to the state $|0\rangle \otimes |0\rangle \otimes |0\rangle = |000\rangle$ then we get

$$\left. \begin{array}{l} |0\rangle \xrightarrow{\boxed{H}} \frac{|0\rangle+|1\rangle}{\sqrt{2}} \\ |0\rangle \xrightarrow{\boxed{H}} \frac{|0\rangle+|1\rangle}{\sqrt{2}} \\ |0\rangle \xrightarrow{\boxed{H}} \frac{|0\rangle+|1\rangle}{\sqrt{2}} \end{array} \right\} = \frac{1}{2^{3/2}} \left\{ \begin{array}{l} |000\rangle + |001\rangle + |010\rangle + |011\rangle \\ + |100\rangle + |101\rangle + |110\rangle + |111\rangle \end{array} \right\}.$$

The resulting state is exactly a superposition of *all* binary string of length 3, and can also be written as

$$\frac{1}{\sqrt{2}}(|0\rangle + |1\rangle) \otimes \frac{1}{\sqrt{2}}(|0\rangle + |1\rangle) \otimes \frac{1}{\sqrt{2}}(|0\rangle + |1\rangle).$$

In general, the tensor product operation $H^{\otimes n}$, which means “apply the Hadamard gate to each of your n qubits”, is known as the **Hadamard transform**, and it maps product states to product states. Like the Hadamard gate in the typical quantum interference circuit, the Hadamard transform opens (and closes) multi-qubit interference.

One final note is on notation, or maybe more a shift of point-of-view. We have just explained how applying the Hadamard transform to n qubits gives us the equally weighted superposition of all binary strings of length n . But rather than writing them as binary strings, we could consider the decimal number represented by each string. This means we switch from considering all binary strings of length n to considering all natural numbers from 0 to $N - 1$, where $N = 2^n$. For example, with $n = 3$ qubits, we could either write

$$\frac{1}{\sqrt{2^n}} \sum_{x \in \{0,1\}^n} |x\rangle$$

or instead switch to the decimal approach with $N = 2^n = 8$ and write

$$\frac{1}{\sqrt{N}} \sum_{x=0}^{N-1} |x\rangle$$

so that we are writing $|7\rangle$ to mean $|111\rangle$, and $|3\rangle$ to mean $|011\rangle$, and $|0\rangle$ to mean $|000\rangle$, and so on.

5.5 Separable or entangled?

“Most” vectors in $\mathcal{H}_a \otimes \mathcal{H}_b$ are **entangled**: they *cannot* be written as product states $|a\rangle \otimes |b\rangle$ with $|a\rangle \in \mathcal{H}_a$ and $|b\rangle \in \mathcal{H}_b$.

In order to see this, let us write any joint state $|\psi\rangle$ of \mathcal{A} and \mathcal{B} in a product basis as

$$\begin{aligned} |\psi\rangle &= \sum_{i,j} c_{ij} |a_i\rangle \otimes |b_j\rangle \\ &= \sum_i |a_i\rangle \otimes \left(\sum_j c_{ij} |b_j\rangle \right) \\ &= \sum_i |a_i\rangle \otimes |\phi_i\rangle \end{aligned} \tag{‡}$$

where the $|\phi_i\rangle = \sum_j c_{ij}|b_j\rangle$ are vectors in \mathcal{H}_B that need not be normalised.

Now, for any *product* state, these vectors have a special form. Indeed, if $|\psi\rangle = |a\rangle \otimes |b\rangle$ then, after expanding the first state in the $|a_i\rangle$ basis, we obtain

$$|\psi\rangle = \sum_i |a_i\rangle \otimes \left(\sum_i \alpha_i |b\rangle \right).$$

This expression has the same form as Equation (‡) with $|\phi_i\rangle = \alpha_i |b\rangle$, i.e. each of the $|\phi_i\rangle$ vectors in this expansion is a multiple of the same vector $|b\rangle$.

Conversely, if $|\phi_i\rangle = \alpha_i |b\rangle$ for all i in Equation (‡), then $|\psi\rangle$ must be a product state. So if we want to identify which joint states are product states and which are not, we simply write the joint state according to Equation (‡) and check if all the vectors $|\phi_i\rangle$ are multiples of a single vector. Needless to say, if we choose the states $|\phi\rangle$ randomly, it is very unlikely that this condition is satisfied, and we almost certainly pick an entangled state. In general, given n qubits, we need $2(2^n - 1)$ real parameters to describe their state vector, but only $2n$ to describe separable states; as n grows larger, $2n$ becomes much *much* smaller than $2(2^n - 1)$.

Even though an entangled state cannot be written as a single tensor product, it can always be written as a *linear combination* of tensor products, since these form a basis.

The Segre embedding.

The problem of deciding whether or not a given state is separable is, in general, a hard problem (i.e. [NP-hard](#)). Because of this, it is interesting to try to understand the notion of separability from different points of view, and it turns out that algebraic geometry yet again has something interesting to say. The theory relies on the notion of [projective space](#), which is a non-trivial topic to try to introduce here, so we do so only briefly, and at a very high speed.

We have repeatedly said that we only really care about state vectors *up to global phase*, i.e. that $|\psi\rangle$ and $|\psi'\rangle$ are “the same” if there exists some θ such that $|\psi\rangle = e^{i\theta}|\psi'\rangle$. Combining this with our unitality requirement (that we want $|\langle\psi|\psi\rangle|^2 = 1$), we are led to studying the equivalence relation

$$v \sim w \iff v = \lambda w \text{ for some } \lambda \in \mathbb{C} \setminus \{0\}$$

on our Hilbert space. Geometrically, this can be understood as *the space of lines through the origin*, i.e. of 1-dimensional subspaces, but the geometry of projective space is a subject that really deserves many *many* pages to delve into, and so we won’t talk about this point of view here.

Algebraically, it turns out that we can describe the space of such equivalence classes using **homogeneous coordinates**. Defining **projective n -space** as

$$\mathbb{P}^n := \mathbb{C}^{n+1} / \sim$$

(where \sim is the equivalence relation defined above), it turns out that points in \mathbb{P}^n are described by coordinates

$$[a_0 : a_1 : \dots : a_n]$$

where $a_i \in \mathbb{C}$ are *not all simultaneously zero* (i.e. there exists at least one $i \in \{0, \dots, n\}$ such that $a_i \neq 0$) and where we impose that

$$[a_0 : a_1 : \dots : a_n] = [\lambda a_0 : \lambda a_1 : \dots : \lambda a_n]$$

for any $\lambda \in \mathbb{C} \setminus \{0\}$.

Why is this useful? Well, given any pure state $\alpha_0|0\rangle + \alpha_1|1\rangle$ of a qubit, we obtain a unique point in \mathbb{P}^1 , namely $[\alpha_0 : \alpha_1]$ (since $|\langle\psi|\psi\rangle|^2 = 1$ tells us that at least one of α_0 and α_1 is non-zero); conversely, given any point $[a_0 : a_1] \in \mathbb{P}^1$,

we can multiply by an appropriate $\lambda \in \mathbb{C} \setminus \{0\}$ to assume that $|a_0|^2 + |a_1|^2 = 1$, and thus obtain a unique (up to global phase) pure state $a_0|0\rangle + a_1|1\rangle$. That is, *points in the (complex) projective line* \mathbb{P}^1 correspond to pure states of a qubit.

Next, we can always express a pure state of two qubits in the form

$$\beta_0|00\rangle + \beta_1|01\rangle + \beta_2|10\rangle + \beta_3|11\rangle$$

and we similarly find a correspondence with points $[z_0 : z_1 : z_2 : z_3]$ in \mathbb{P}^3 (given, in one direction, by setting $z_i := \beta_i$).

What is of interest to us here is a particular map known as the **Segre embedding**:

$$\begin{aligned}\sigma: \mathbb{P}^1 \times \mathbb{P}^1 &\longrightarrow \mathbb{P}^3 \\ ([a_0 : a_1], [b_0 : b_n]) &\longmapsto [a_0 b_0 : a_0 b_1 : a_1 b_0 : a_1 b_1].\end{aligned}$$

First of all, one needs to check that this does indeed give a well defined function (i.e. that the resulting coordinate always has at least one non-zero component, and that it is invariant under multiplication by a non-zero scalar $\lambda \in \mathbb{C} \setminus \{0\}$). But it turns out that, not only is this a well defined function, but it is actually a “geometric” function, in that it respects the “geometric structure” of projective space. We won’t concern ourselves here with what that means, but we note that it is even more well behaved than this: as its name suggests, it actually gives an **embedding** (i.e. a “geometric” injection) of the 2-dimensional space $\mathbb{P}^1 \times \mathbb{P}^1$ into the 3-dimensional space \mathbb{P}^3 .

The image of the Segre embedding is called the **Segre variety**, and you can check that it is given by the set of points

$$\Sigma := \text{Im}(\sigma) = \{[z_0 : z_1 : z_2 : z_3] \in \mathbb{P}^3 \mid z_0 z_3 - z_1 z_2 = 0\}$$

(in algebraic-geometry language, it is the **zero-locus** of a single polynomial).

Now here is the punchline to all this geometric meandering: *a state $|\phi\rangle$ of two qubits is separable if and only if its corresponding point in \mathbb{P}^3 lies in the Segre variety Σ .*

For more, see e.g. Cirici, Salvadó, and Taron’s “Characterization of quantum entanglement via a hypercube of Segre embeddings”, arXiv:[2008.09583](#)).

Quantum entanglement is one of the most fascinating aspects of quantum theory. We will now explore some of its computational implications.

5.6 Controlled-NOT

How do entangled states arise in real physical situations? The short answer is that *entanglement is the result of interactions*. It is easy to see that tensor product operations $U_1 \otimes \dots \otimes U_n$ map product states to product states:

$$\left. \begin{array}{c} |\psi_1\rangle \xrightarrow{\boxed{U_1}} |\psi'_1\rangle \\ \vdots \\ |\psi_n\rangle \xrightarrow{\boxed{U_n}} |\psi'_n\rangle \end{array} \right\} |\psi'_1\rangle \otimes \dots \otimes |\psi'_n\rangle$$

and so any collection of separable qubits remains separable. As soon as qubits start interacting with one another, however, they become entangled, and things start to get really interesting. We will describe interactions that cannot be written as tensor products of unitary operations on individual qubits.

The most popular two-qubit entangling gate is the **controlled-NOT** (or c-NOT), also known as the **controlled-X** gate. The gate acts on two qubits: it flips the second qubit (referred to as the **target**) if the first qubit (referred to as the **control**) is $|1\rangle$, and does nothing if the control qubit is $|0\rangle$. In the standard basis $\{|00\rangle, |01\rangle, |10\rangle, |11\rangle\}$, it is represented by the following unitary matrix:

$$\text{Controlled-NOT: } \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 1 & 0 \end{bmatrix}$$

Here, $X \equiv \sigma_x$ refers to the Pauli operator that implements the bit-flip.

We represent the c-NOT gate in circuit notation as shown in Figure 5.1.

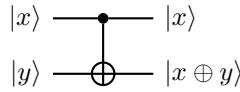


Figure 5.1: Where $x, y \in \{0, 1\}$, and \oplus denotes XOR, or addition modulo 2.

Note that this gate does not admit any tensor-product decomposition, but can be written as a sum of tensor products:

$$\text{c-NOT} = |0\rangle\langle 0| \otimes \mathbf{1} + |1\rangle\langle 1| \otimes X$$

(where X is the Pauli bit-flip operation).

The c-NOT gate lets us do many interesting things, and can act in a rather deceptive way. Let us now study some of these things.

5.7 Bell states

We start with the generation of entanglement. Here is a simple circuit that demonstrates the entangling power of c-NOT:

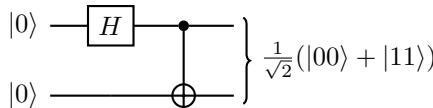
Make sure that you understand how the Dirac notation is used here. More generally, think why

$$|0\rangle\langle 0| \otimes A + |1\rangle\langle 1| \otimes B$$

means “if the first qubit is in state $|0\rangle$ then apply A to the second one, and if the first qubit is in state $|1\rangle$ then apply B to the second one”. What happens if the first qubit is in a superposition of $|0\rangle$ and $|1\rangle$?

John Stewart Bell (1928–1990) was a Northern Irish physicist.

Circuit. (Generating entanglement).



In this circuit, the separable input $|0\rangle|0\rangle$ evolves as

$$\begin{aligned} |0\rangle|0\rangle &\xrightarrow{H} \frac{1}{\sqrt{2}}(|0\rangle + |1\rangle)|0\rangle \\ &= \frac{1}{\sqrt{2}}|0\rangle|0\rangle + \frac{1}{\sqrt{2}}|1\rangle|0\rangle \\ &\xrightarrow{\text{c-NOT}} \frac{1}{\sqrt{2}}|0\rangle|0\rangle + \frac{1}{\sqrt{2}}|1\rangle|1\rangle \end{aligned}$$

resulting in the entangled output $\frac{1}{\sqrt{2}}(|00\rangle + |11\rangle)$. In fact, this circuit implements the unitary operation which maps the standard computational basis into the four entangled states, known as the **Bell states**.

The **Bell states** $|\psi_{ij}\rangle$ are those generated by the above circuit:

$$|00\rangle \mapsto |\psi_{00}\rangle := \frac{1}{\sqrt{2}}(|00\rangle + |11\rangle)$$

$$|01\rangle \mapsto |\psi_{01}\rangle := \frac{1}{\sqrt{2}}(|01\rangle + |10\rangle)$$

$$|10\rangle \mapsto |\psi_{10}\rangle := \frac{1}{\sqrt{2}}(|00\rangle - |11\rangle)$$

$$|11\rangle \mapsto |\psi_{11}\rangle := \frac{1}{\sqrt{2}}(|01\rangle - |10\rangle)$$

The more standard notation for these states, however, is the following:

$$\Phi^+ := |\psi_{00}\rangle$$

$$\Psi^+ := |\psi_{01}\rangle$$

$$\Phi^- := |\psi_{10}\rangle$$

$$\Psi^- := |\psi_{11}\rangle$$

(and this is the notation that we will use from now on).

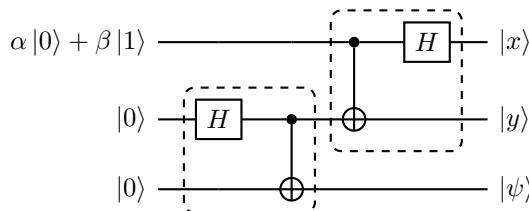
The Bell states form an orthonormal basis in the Hilbert space $\mathcal{H}_1 \otimes \mathcal{H}_2$ of two qubits. We can perform measurements in the Bell basis: the easiest way to do it in practice is to “rotate” the Bell basis to the standard basis, and then perform the measurement in the standard basis. Indeed, if we reverse the circuit (running it from right to left), then we get a circuit which maps the Bell state $|\psi_{ij}\rangle$ to the corresponding state $|ij\rangle$ in the standard basis. This unitary mapping allows us to “implement” the projections on Bell states by applying the reversed circuit followed by the usual qubit-by-qubit measurement in the standard basis.

The Bell states are said to be **maximally entangled**, since their reduced density operators are maximally mixed (a notion that we will define in Section 8.3). Roughly, this means that the outcomes of *any* measurement performed on them are completely random. This property — having maximal entropy (in some sense) — makes the Bell states incredibly useful for many applications, and we shall see some of them now.

5.8 Quantum teleportation

A wonderful fact, that sounds more like science fiction than actual science, is the following: *an unknown quantum state can be teleported from one location to another*. Consider the following circuit, which is built from a Bell state generator followed by an “offset” inverse Bell state generator:

Circuit. (Quantum teleportation).



For any state $|\psi\rangle$ of two qubits, the amplitude $\langle ij|U^\dagger|\psi\rangle$ can be written as $\langle ij|U^\dagger|\psi\rangle$, where U^\dagger is such that $|\psi_{ij}\rangle = U|ij\rangle$.

Divide et impera, or “divide and conquer”: a good approach to solving problems in mathematics (and in life). Start with the smaller circuits in the dashed boxes, which we have just seen introduced above.

The first input qubit (counting from the top) is in some arbitrary state. After the action of the part of the circuit in the first dashed box (counting from the left), the state of the three qubits reads

$$(\alpha|0\rangle + \beta|1\rangle)(|00\rangle + |11\rangle).$$

By regrouping the terms, but keeping the qubits in the same order, this state can be written as the sum

$$\begin{aligned} &(|00\rangle + |11\rangle) \otimes (\alpha|0\rangle + \beta|1\rangle) \\ &+ (|01\rangle + |10\rangle) \otimes (\alpha|1\rangle + \beta|0\rangle) \\ &+ (|00\rangle - |11\rangle) \otimes (\alpha|0\rangle - \beta|1\rangle) \\ &+ (|01\rangle - |10\rangle) \otimes (\alpha|1\rangle - \beta|0\rangle). \end{aligned}$$

Then the part of the circuit in the second dashed box maps the four Bell states of the first two qubits to the corresponding states from the computational basis:

$$\begin{aligned} &|00\rangle \otimes (\alpha|0\rangle + \beta|1\rangle) \\ &+ |01\rangle \otimes (\alpha|1\rangle + \beta|0\rangle) \\ &+ |10\rangle \otimes (\alpha|0\rangle - \beta|1\rangle) \\ &+ |11\rangle \otimes (\alpha|1\rangle - \beta|0\rangle). \end{aligned}$$

Upon performing the standard measurement and learning the values of x and y , we choose one of the four following transformations depending on these values:

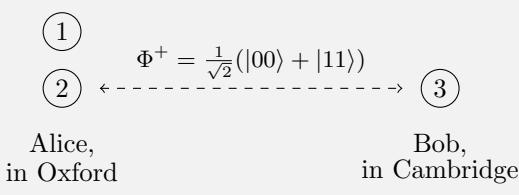
$$\begin{array}{ll} 00 \mapsto 1 & 01 \mapsto X \\ 10 \mapsto Z & 11 \mapsto ZX \end{array} \quad (*)$$

(e.g. if $x = 0$ and $y = 1$, then we choose X). We then apply this transformation to the third qubit, which restores the original state of the first qubit.

If you understand how this circuit works, then you are ready for quantum teleportation. Here is a dramatic version.

You can play around with this on the [Quantum Flytrap Virtual Lab](#).

Suppose that three qubits, which all look very similar, are initially in the possession of an absent-minded Oxford student, Alice. The first qubit is in a precious quantum state and this state is needed urgently for an experiment in Cambridge. The other two qubits are entangled, in the $\Phi^+ = |\psi_{00}\rangle$ state. Alice's colleague, Bob, pops in to collect the qubit. Once he is gone, Alice realises that, by mistake, she gave him not the first but the third qubit: the one which is entangled with the second qubit.



The situation seems to be hopeless — Alice does not know the quantum state of the first qubit, and Bob is now miles away and her communication with him is limited to few bits. However, Alice and Bob are both very clever and they both diligently attended their “Introduction to Quantum Information Science” classes. Can Alice rectify her mistake and save Cambridge science?

...

Of course: Alice can teleport the state of the first qubit! She performs the Bell measurement on the first two qubits, which gives her two binary digits, x and y . She then broadcasts x and y to Bob, who chooses the corresponding transformation, as in Equation (⊗), performs it, and recovers the original state.

This raises a natural “philosophical” question: what do we really *mean* by teleportation? A key part of this question is understanding what happens to our original qubit when we teleport it. Note that the actual physical electron (or whatever implementation of qubits we are using) does not suddenly move through space — what is teleported is the *state* of the qubit, but the argument can be made that if two qubits are entirely indistinguishable from one another by any measurements that we can make, then they really are “the same” in every way that matters, and so the qubit which now has the original qubit’s state “is the same as” the original qubit. As it turns out, this process necessarily *destroys* the original qubit’s state, as we now explain.

Teleportation experiments and verification.

The first actual teleportation experiment was successfully achieved in 1997 (arXiv:[quant-ph/9710013](#)); in 2012 a record distance was set: an entangled photon pair was used to teleport a state 143 kilometres/88 miles (arXiv:[1205.3909](#)); in 2017, successful ground-to-satellite teleportation was achieved (arXiv:[1707.00934](#)). This is not science fiction!

But there is a fundamental question to ask: if the original state is destroyed, then how can we really verify that teleportation has taken place? We can’t compare the purportedly teleported state to the original one! The answer to this involves certain **no-go theorems** and statistical methods, where we can show that classical physics gives some strict upper bound on a certain fidelity, but which is clearly surpassed by these physical experiments. We will better explain the ideas behind these sorts of arguments later on, in Chapter 6, when we introduce **Bell’s theorem**.

5.9 No-cloning, and other no-go theorems

Let us now look at something that the controlled-NOT *seems* to be doing but, in fact, *isn’t*. It is easy to see that the c-NOT can copy the bit value of the first qubit:

$$|x\rangle|0\rangle \xrightarrow{\text{c-NOT}} |x\rangle|x\rangle \quad (\text{for } x = 0, 1)$$

so one might suppose that this gate could also be used to copy superpositions, such as $|\psi\rangle = \alpha|0\rangle + \beta|1\rangle$, so that

$$|\psi\rangle|0\rangle \xrightarrow{\text{c-NOT}} |\psi\rangle|\psi\rangle$$

for any $|\psi\rangle$. But this is not true!

The unitarity of the c-NOT means that it turns *superpositions* in the control qubit into *entanglement* of the control and the target: if the control qubit is in a superposition state $|\psi\rangle = \alpha|0\rangle + \beta|1\rangle$ (with $\alpha, \beta \neq 0$), and the target is in $|0\rangle$, then the c-NOT gate generates the entangled state

$$(\alpha|0\rangle + \beta|1\rangle)|0\rangle \xrightarrow{\text{c-NOT}} \alpha|00\rangle + \beta|11\rangle.$$

In fact, it is *impossible* to clone an unknown quantum state, and we can prove this!

To prove this via contradiction, let us assume that we *could* build a universal quantum cloner, and then take any two normalised states $|\psi\rangle$ and $|\phi\rangle$ that are *non-identical* (i.e. $|\langle\psi|\phi\rangle| \neq 1$) and *non-orthogonal* (i.e. $\langle\psi|\phi\rangle \neq 0$). If we then run our hypothetical cloning machine we get

$$\begin{aligned} |\psi\rangle|0\rangle|W\rangle &\longmapsto |\psi\rangle|\psi\rangle|W'\rangle \\ |\phi\rangle|0\rangle|W\rangle &\longmapsto |\phi\rangle|\phi\rangle|W''\rangle \end{aligned}$$

where the third system, initially in state $|W\rangle$, represents everything else (say, the internal state of the cloning machine). For this transformation to be unitary, it must preserve the inner product, and so we require that

$$\langle\psi|\phi\rangle = \langle\psi|\phi\rangle^2\langle W'|W''\rangle$$

which can only be satisfied if $|\langle\psi|\phi\rangle|$ is equal to 1 or 0, but this contradicts our assumptions!

Thus, states of qubits, unlike states of classical bits, cannot be faithfully cloned. Note that, in quantum teleportation, the original state must therefore be *destroyed*, since otherwise we would be producing a clone of an unknown quantum state. The no-cloning property of quantum states leads to interesting applications, of which quantum cryptography is one.

The no-cloning theorem. Universal quantum cloners are *impossible*.

Approximate quantum cloning.

This section is not yet finished.

The no-cloning theorem is one of many so-called “no-go” theorems in quantum information. We won’t look at all of them in depth, but it’s worth mentioning them here and giving a very rough idea of what each one says.

- **No-teleportation.** *An arbitrary quantum state cannot be entirely expressed with classical information. In other words, the process of converting quantum information to classical information cannot be reversed: classical channels cannot transmit quantum information.*

This can be seen as a consequence of no-cloning: if we were able to turn a quantum state into classical information and then back again, we could simply clone the classical information and then get a cloned copy of our quantum state.

The name is a bit confusing, because we have just seen that *quantum* teleportation is possible through the use of entanglement, but it refers to the idea of *classical* teleportation of quantum states.

Note that the “converse” to this is possible though: if we start with some *classical* information then we can convert it to quantum information and then back again perfectly fine (for example, using the fact that orthogonal states can be perfectly distinguished).

- **No-broadcasting.** *Given a single copy of a quantum state, it cannot be shared with two or more parties.*

This is an even more direct consequence of no-cloning: if we can’t copy a state, then we have no way of sharing it with multiple people. However, the real technical statement of this theorem involves **non-pure states**, which require the language of **density operators** to talk about — something that we will not see until Chapter 8.

One particularly unexpected detail here is that the theorem is no longer true if we're provided with more than one copy of the state to start with. For example, in a process known as **superbroadcasting**, given four copies of an input state we can actually broadcast six copies!

- **No-deleting.** *Given two copies of an arbitrary quantum state, it is impossible to delete one.*

You might hear people saying that the fact that we require our quantum operations to be unitary is to do with **reversibility**, and so there is a general pattern in quantum theory where theorems will have **time-dual** versions, giving by taking the same theorem but imagining that time goes in the opposite direction. No-deleting is the time dual of no-cloning, and whereas the latter tells us that quantum states are pretty delicate, the former tells us that they are also in some sense rather robust.

We might as well state this theorem a bit more precisely, because we have seen almost all of the necessary definitions already: given a qubit in an unknown state $|\psi\rangle$, there is no **isometry** V (Section 9.3) such that

$$V : |\psi\rangle|\psi\rangle|W\rangle \longmapsto |\psi\rangle|0\rangle|W'\rangle$$

with $|W'\rangle$ being independent of $|\psi\rangle$. Just as for no-cloning, we can of course delete *some qubits* (for example those in orthogonal states, since these behave a lot like classical bits), but there is no V that works universally, for *any* arbitrary state $|\psi\rangle$.

- **No-communication.** *An entangled state cannot be used to transmit information by measurement of a subsystem.*

We talk about this theorem in the context of a worked example in Exercise 5.14.3, and we delve into the details when we talk about **Bell tests** in Chapter 6, but it is basically the answer to Einstein's worry about "spooky action at a distance" that we mentioned back in Section 5.1: the seemingly infinitely fast sending of information between entangled qubits cannot actually send any meaningful information, but only purely random bits.

This theorem is actually stronger than no-cloning, in that we can prove no-cloning from no-communication.

Yet again we see another example of how the quantum whole is much greater than the sum of its parts: no-teleportation says that classical channels alone cannot send quantum information; no-communication says that entanglement and measurement alone cannot send quantum information; the quantum teleportation protocol of Section 5.8 says that you *can* send quantum information *if you combine both methods together*.

- **No-hiding.** *Quantum information cannot be lost, even through decoherence.*

This theorem is related to no-deletion, in that it shows the robustness of quantum states. In Chapter 13 we will study the notion of **decoherence**, which is sort of like "quantum noise", and is one of the main problems faced when actually trying to design and build quantum computers in reality. The no-hiding theorem says that, when quantum information is "lost" through decoherence, it actually merely moves into the subspace corresponding to the environment — we might have lost it, but nature hasn't.

This is shown in D'Ariano, Macchiavello, and Perinotti's "Superbroadcasting of mixed states", arXiv:[quant-ph/0506251](https://arxiv.org/abs/quant-ph/0506251).

In fact, we'll talk a bit about reversibility of computation in Section 10.1.

5.10 Controlled-phase and controlled-U

Needless to say, not everything is about the controlled-NOT gate. Another common two-qubit gate is the **controlled-phase** gate, denoted $c-P_\varphi$.

This theorem is of particular interest to physicists studying black holes, since it leads to the **black hole information paradox**.

Controlled-phase:

$$\left[\begin{array}{cc|cc} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ \hline 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & e^{i\varphi} \end{array} \right]$$

We can also represent the $c-P_\varphi$ gate using the circuit notation, as in Figure 5.2.

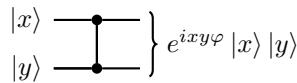
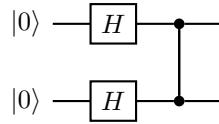


Figure 5.2: Where $x, y \in \{0, 1\}$.

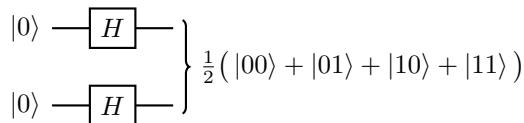
Again, the matrix is written in the computational basis $\{|00\rangle, |01\rangle, |10\rangle, |11\rangle\}$. If we do not specify the phase then we usually assume that $\varphi = \pi$, in which case we call this operation the **controlled-Z gate**, which acts as $|0\rangle\langle 0| \otimes \mathbf{1} + |1\rangle\langle 1| \otimes Z$. Here Z refers again to the Pauli phase-flip $\sigma_z \equiv Z$ operation.

In order to see the entangling power of the controlled-phase shift gate, consider the following circuit.

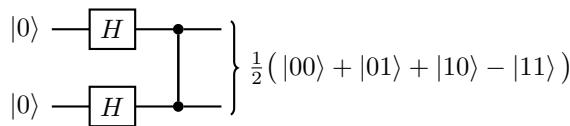
Circuit. (Generating entanglement, again).



In this circuit, first the two Hadamard gates prepare the equally-weighted superposition of all states from the computational basis



and then the controlled-Z operation flips the sign in front of $|11\rangle$



which results in an entangled state.

In fact, both $c\text{-NOT}$ and $c\text{-}P_\varphi$ are specific examples of the more general construction of a **controlled-U gate**:

$$c\text{-}U = |0\rangle\langle 0| \otimes \mathbf{1} + |1\rangle\langle 1| \otimes U$$

where U is an arbitrary single-qubit unitary transformation U .

Controlled- U :

$$\left[\begin{array}{cc|cc} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ \hline 0 & 0 & & U \\ 0 & 0 & & \end{array} \right]$$

We can also represent the $c\text{-}U$ gate using the circuit notation, as in Figure 5.3.

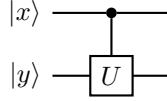


Figure 5.3: The controlled- U gate, where $x, y \in \{0, 1\}$.

We can go even further and consider a more general unitary operation: the two-qubit **x -controlled- U gate**:

$$\sum_x |x\rangle\langle x| \otimes U_x \equiv |0\rangle\langle 0| \otimes U_0 + |1\rangle\langle 1| \otimes U_1$$

where each U_x is a unitary transformation that is applied to the second qubit only if the first one is in state $|x\rangle$. In general, an x -controlled- U gate can be defined on two registers of arbitrary size n and m , with $x \in \{0, 1\}^n$ and the U_x being $(2^m \times 2^m)$ unitary matrices acting on the second register.

5.11 Universality, revisited

We will come across few more gates in this course, but at this stage you already know all the elementary unitary operations that are needed to construct any unitary operation on any number of qubits:

- the Hadamard gate,
- all phase gates, and
- the $c\text{-NOT}$

These gates form a **universal set of gates**: with $\mathcal{O}(4^n n)$ of these gates, we can construct any n -qubit unitary operation. We should mention that there are many different universal sets of gates. In fact, almost any gate that can entangle two qubits can be used as a universal gate.

We are particularly interested in any *finite* universal set of gates that can approximate any unitary operation on n qubits with arbitrary precision. The price to pay is the number of gates — better precision requires more gates.

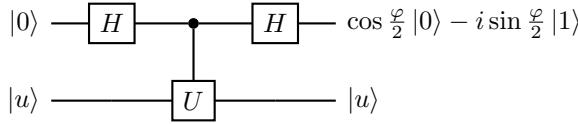
5.12 Phase kick-back

Before moving on, we first describe a simple yet omnipresent “trick” — an unusual way of introducing phase shifts that will be essential for our analysis of quantum algorithms. Consider the following circuit.

Recall the big- \mathcal{O} asymptotic notation introduced in Exercise 1.11.7: given a *positive* function $f(n)$, we write $\mathcal{O}(f(n))$ to mean “bounded above by $c f(n)$ for some constant $c > 0$ (for sufficiently large n)”. For example, $15n^2 + 4n + 7$ is $\mathcal{O}(n^2)$.

One particular example that we will see again is the Hadamard, $c\text{-NOT}$, and $T = P_{\pi/4}$.

Circuit. (Controlled- U interference).



where $|u\rangle$ is an eigenstate of U , so that $U|u\rangle = e^{i\varphi}|u\rangle$ for some φ .

This should look familiar: it is the usual interference circuit, but with the phase gate replaced by a controlled- U gate, which will mimic the phase gate, as we shall soon see. Note that the second qubit is prepared in state $|u\rangle$, which is *required to be an eigenstate of U* . The circuit effects the following sequence of transformations:

$$\begin{aligned} |0\rangle|u\rangle &\xrightarrow{H} (|0\rangle + |1\rangle)|u\rangle \\ &= |0\rangle|u\rangle + |1\rangle|u\rangle \\ &\xrightarrow{\text{c-}U} |0\rangle|u\rangle + |1\rangle U|u\rangle \\ &= |0\rangle|u\rangle + e^{i\varphi}|1\rangle|u\rangle \\ &= (|0\rangle + e^{i\varphi}|1\rangle)|u\rangle \\ &\xrightarrow{H} \left(\cos\frac{\varphi}{2}|0\rangle - i\sin\frac{\varphi}{2}|1\rangle\right)|u\rangle. \end{aligned}$$

Omitting, as per usual, the normalisation factors.

Note that the second qubit does *not* get entangled with the first one: it remains in its original state $|u\rangle$. However, the interaction between the two qubits introduces a phase shift on the first qubit. This may look like an unnecessarily complicated way of introducing phase shifts, but, as we shall soon see, this is how quantum computers do it. Here is a preview of things to come.

Consider the following x -controlled- U operation:

$$\begin{bmatrix} \mathbf{1} & 0 & 0 & 0 \\ 0 & \mathbf{1} & 0 & 0 \\ 0 & 0 & \mathbf{1} & 0 \\ 0 & 0 & 0 & X \end{bmatrix} = \begin{aligned} &|00\rangle\langle 00| \otimes \mathbf{1} \\ &+ |01\rangle\langle 01| \otimes \mathbf{1} \\ &+ |10\rangle\langle 10| \otimes \mathbf{1} \\ &+ |11\rangle\langle 11| \otimes X. \end{aligned}$$

The first register is of size 2 (corresponding to the top-left 2×2 block, which is simply the identity matrix), and the second register is of size 1 (corresponding to the two bottom-right 1×1 blocks, namely an identity matrix and X). If the first register is prepared in state $|11\rangle$, then the qubit in the second register is flipped (by the Pauli bit-flip X); otherwise, nothing happens.

This unitary operation is a quantum version of the Boolean function evaluation: it corresponds to the Boolean function

$$\begin{aligned} f: \{0, 1\}^2 &\longrightarrow \{0, 1\} \\ 00 &\longmapsto 0 \\ 01 &\longmapsto 0 \\ 10 &\longmapsto 0 \\ 11 &\longmapsto 1. \end{aligned}$$

If $f(x) = 1$, then we flip the bit value in the second register (with operation X); if $f(x) = 0$, then we do nothing.

Now, prepare the qubit in the second register in state $|0\rangle - |1\rangle$, which is an eigenstate of X with eigenvalue $e^{\pi i} = -1$. So whenever X is applied to the second reg-

ister, the phase factor -1 appears in front of the corresponding term in the first register. If we prepare the first register in the superposition $|00\rangle + |01\rangle + |10\rangle + |11\rangle$ then the result of applying the above x -controlled- U operation is the entangled state $|00\rangle + |01\rangle + |10\rangle - |11\rangle$. That is, *the phase kick-back mechanism introduces a relative phase in the equally-weighted superposition of all binary strings of length two.*

Phase kick-back is how we control quantum interference in quantum computation.

We will return to this topic later on in Section 10.2, when we discuss quantum evaluation of Boolean functions and quantum algorithms.

5.13 Density operators, and other things to come

The existence of entangled states leads to an obvious question: if we cannot attribute a state vector to an individual qubit, then how can we describe its quantum state? In the next few chapters we will see that, when we limit our attentions to a part of a larger system, states are not represented by vectors, measurements are not described by orthogonal projections, and evolution is not unitary. As a spoiler, here is a dictionary of some of the new concepts that will soon be introduced:

state vectors	\leadsto	density operators
orthogonal projectors	\leadsto	positive operator-valued measures
unitary evolutions	\leadsto	completely-positive trace-preserving maps

5.14 Remarks and exercises

5.14.1 Why qubits, subsystems, and entanglement?

One question that is rather natural to ask at this point is the following:

If entanglement is so fragile and difficult to control, then why bother? Why not perform your computations in one singly physical system that has as many quantum states as we normally have labels for the states of qubits? Then we could label these quantum states in the same way as we normally label the qubits, and give them computational meaning.

This suggestion, although possible, gives a very inefficient way of representing data, known as the **unary encoding**. For serious computations, we *need* subsystems. Here is why.

Suppose you have n physical objects, and each object has k distinguishable states. If you can access each object *separately* and put it into any of the k states, then, with only n operations, you can prepare any of the k^n different configurations of the combined systems. Without any loss of generality, let us take $k = 2$ and refer to each object of this type as a **physical bit**. We label the two states of a physical bit as 0 and 1. So any collection of n physical bits can be prepared in 2^n different configurations, which can be used to store up to 2^n numbers (or binary strings, or messages, or however you want to interpret these things). In order to represent numbers from 0 to $N - 1$ we just have to choose n such that $2^n \geq N$.

Suppose the two states in the physical bit are separated by the energy difference $\Delta_E > 0$, i.e. that it costs Δ_E units of energy to switch a physical bit from one state to the other. Then a preparation of any particular configuration will cost no more than $E = n\Delta_E = (\log_2 N)\Delta_E$ units of energy.

In contrast, if we choose to encode N configurations into one chunk of matter, say, into the first N energy states of a single harmonic oscillator with the same energy separation Δ_E between states, then, in the worst case (i.e. going from the ground state 0 to the most excited state N) one has to use $E = N\Delta_E$ units of energy. For large N this gives an exponential gap in the energy expenditure between the binary encoding using physical bits, and the unary encoding using energy levels of harmonic oscillators: $(\log_2 N)\Delta_E$ vs $N\Delta_E$.

Of course, you might try to switch to a different choice of realisation for the unary encoding, such as a quantum system that has a finite spread in the energy spectrum. For example, by operating on the energy states of the hydrogen atom, you can encode any number from 0 to $N - 1$, and we are guaranteed not to spend more than $E_{\max} = 13.6 \text{ eV}$ (otherwise the atom is ionised). The snag is that, in this case, some of the electronic states will be separated by an energy difference to the order of E_{\max}/N , and to drive the system selectively from one state to another one has to tune into the frequency $E_{\max}/\hbar N$, which requires a sufficiently long **wave packet** in order for the frequency to be well defined, and consequently the interaction time is of order $N(\hbar/E_{\max})$.

That is, we spend *less energy*, but the trade off is that we have to spend *more time*.

It turns out that whichever way we try to represent the number N in the unary encoding (i.e. using N different states of a single chunk of matter), we end up depleting our physical resources (such as energy or time, or even space) at a much greater rate than in the case when we use subsystems. This plausibility argument indicates that, for efficient processing of information, the system must be divided into subsystems — for example, into physical bits.

5.14.2 Entangled or not?

Let a joint state of \mathcal{A} and \mathcal{B} be written in a product basis as

$$|\psi\rangle = \sum_{i,j} c_{ij} |a_i\rangle \otimes |b_j\rangle.$$

Assume that $\mathcal{H}_{\mathcal{A}}$ and $\mathcal{H}_{\mathcal{B}}$ are of equal dimension.

1. Show that, if $|\psi\rangle$ is a product state, then $\det(c_{ij}) = 0$.
2. Show that the converse ($\det(c_{ij}) = 0 \implies |\psi\rangle = |a\rangle|b\rangle$) holds only for qubits. Explain why.
3. Deduce that the state

$$\frac{1}{2}(|00\rangle + |01\rangle + |10\rangle + (-1)^k|11\rangle)$$

is entangled for odd values of k and unentangled for even values of k . Express the latter case explicitly as a product state.

5.14.3 Instantaneous communication

There is a lot of interesting physics behind the innocuous-looking mathematical statement of Exercise 5.14.2. For example, think again about the state $(|00\rangle + |11\rangle)/\sqrt{2}$. What happens if you measure just the first qubit? It is equally likely that you get $|0\rangle$ or $|1\rangle$, right? But after your measurement the two qubits are either in state $|00\rangle$ or in $|11\rangle$, i.e. they show the same bit value. Now, why might that be disturbing? Well, imagine the second qubit to be light-years away from the first one. It seems that the

For simplicity here, we're assuming that $N = 2^n$.

measurement of the first qubit affects the second qubit right away, which seems to imply faster-than-light communication! This is what Einstein called “spooky action at a distance” in his 1947 letter to Max Born.

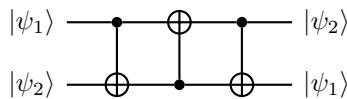
But can you actually use this effect to send a message faster than light? What would happen if you tried?

Hopefully you can see that it would not work, since the result of the measurement is random: you cannot choose the bit value you want to send. We shall return to this, and other related phenomena, later on — it is not at all a lost cause!

5.14.4 SWAP circuit

Show that, for any states $|\psi_1\rangle$ and $|\psi_2\rangle$ of two qubits, the circuit below implements the SWAP operation $|\psi_1\rangle|\psi_2\rangle \mapsto |\psi_2\rangle|\psi_1\rangle$.

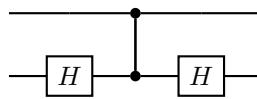
Circuit. (Swapping).



5.14.5 Controlled-NOT circuit

Show that the circuit below gives another implementation of the controlled-NOT gate.

Circuit. (Controlled-NOT, again).

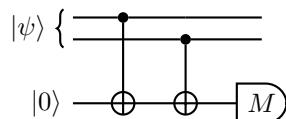


5.14.6 Measuring with controlled-NOT

The controlled-NOT gate can act as a measurement gate: if you prepare the target in state $|0\rangle$ then the gate acts as $|x\rangle|0\rangle \mapsto |x\rangle|x\rangle$, and so the target learns the bit value of the control qubit. If you wish, you can think about a subsequent measurement of the target qubit in the computational basis as an observer learning about the bit value of the control qubit.

Take a look at the circuit below, where M stands for measurement in the standard basis.

Circuit. (?).



Now assume that the top two qubits are in the state

$$|\psi\rangle = \frac{1}{\sqrt{3}}(|01\rangle - |10\rangle + i|11\rangle).$$

The measurement M gives two possible outcomes: 0 and 1. What are the probabilities of each outcome, and what is the post-measurement state in each case?

What is this circuit actually measuring?

5.14.7 Arbitrary controlled-U on two qubits

Recall Section 3.5: any unitary operation U on a single qubit can be expressed as

$$U = B^\dagger X B A^\dagger X A$$

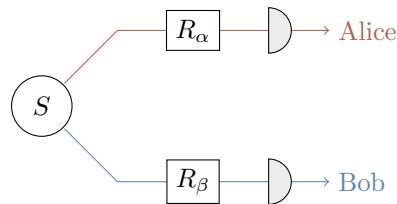
for some unitaries A and B , where $X \equiv \sigma_x$ is the Pauli bit-flip operator.

Suppose that you can implement any single-qubit gate, and that you have a bunch of controlled-NOT gates at your disposal. How would you implement any controlled- U operation on two qubits?

5.14.8 Entangled qubits

Two entangled qubits in the state $\frac{1}{\sqrt{2}}(|00\rangle + |11\rangle)$ are generated by some source S . One qubit is sent to Alice, and one to Bob, who then both perform measurements in the computational basis.

1. What is the probability that Alice and Bob will register identical results? Can any correlations they observe be used for instantaneous communication?
2. Prior to the measurements in the computational basis, Alice and Bob apply unitary operations R_α and R_β (respectively), for some fixed values $\alpha, \beta \in \mathbb{R}$, to their respective qubits:



where the gate R_θ is defined by its action on the basis states:

$$\begin{aligned} |0\rangle &\mapsto \cos \theta |0\rangle + \sin \theta |1\rangle \\ |1\rangle &\mapsto -\sin \theta |0\rangle + \cos \theta |1\rangle. \end{aligned}$$

Show that the state of the two qubits prior to the measurements is

$$\begin{aligned} &\frac{1}{\sqrt{2}} \cos(\alpha - \beta) (|00\rangle + |11\rangle) \\ &- \frac{1}{\sqrt{2}} \sin(\alpha - \beta) (|01\rangle - |10\rangle). \end{aligned}$$

3. What is the probability that Alice and Bob's outcomes are identical?
4. What is the geometric interpretation of the operator R_θ ?

5.14.9 Quantum dense coding

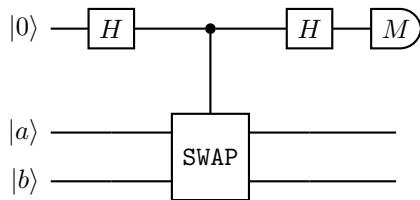
This section is not yet finished.

5.14.10 Playing with conditional unitaries

The swap gate SWAP on two qubits is defined first on product vectors by SWAP : $|a\rangle|b\rangle \mapsto |b\rangle|a\rangle$ and then extended to sums of product vectors by linearity (see Exercise 5.14.4).

1. Show that the four Bell states $\frac{1}{\sqrt{2}}(|00\rangle \pm |11\rangle)$ and $\frac{1}{\sqrt{2}}(|01\rangle \pm |10\rangle)$ are eigenvectors of SWAP that form an orthonormal basis in the Hilbert space associated to two qubits. Which Bell states span the *symmetric subspace* (i.e. the space spanned by all eigenvectors with eigenvalue 1), and which the *antisymmetric* one (i.e. that spanned by eigenvectors with eigenvalue -1)? Can SWAP have any eigenvalues apart from ± 1 ?
2. Show that $P_{\pm} = \frac{1}{2}(\mathbf{1} \pm \text{SWAP})$ are two orthogonal projectors which form the decomposition of the identity and project onto the symmetric and antisymmetric subspaces. Decompose the state vector $|a\rangle|b\rangle$ of two qubits into symmetric and antisymmetric components.
3. Consider the quantum circuit below, composed of two Hadamard gates, one controlled-SWAP operation (also known as the **controlled-swap**, or **Fredkin gate**), and the measurement M in the computational basis. Suppose that the state vectors $|a\rangle$ and $|b\rangle$ are normalised but *not* orthogonal to one another. Step through the execution of this network, writing down the quantum states of the three qubits after each of the four computational steps. What are the probabilities of observing 0 or 1 when the measurement M is finally performed?

Circuit. (Symmetric and antisymmetric projection).



4. Explain why this quantum network implements projections on the symmetric and antisymmetric subspaces of the two qubits.
5. Two qubits are transmitted through a quantum channel which applies the same randomly chosen unitary operation U to each of them, i.e. $U \otimes U$. Show that the symmetric and antisymmetric subspaces are invariant under this operation.
6. Polarised photons are transmitted through an optical fibre. Due to the variation of the refractive index along the fibre, the polarisation of each photon is rotated by the same unknown angle. This makes communication based on polarisation encoding unreliable. However, if you are able to prepare *any* polarisation state of the two photons then you can still use the channel to communicate without any errors — how?

5.14.11 Tensor products in components

In our discussion of tensor products we have so far taken a rather abstract approach. There are, however, situations in which we have to put numbers in, and write tensor products of vectors and matrices explicitly. For example, here is the standard basis of two qubits written explicitly as column vectors:

We always use the lexicographic order $00 < 01 < 10 < 11$.

$$|00\rangle \equiv |0\rangle \otimes |0\rangle = \begin{bmatrix} 1 \\ 0 \end{bmatrix} \otimes \begin{bmatrix} 1 \\ 0 \end{bmatrix} = \begin{bmatrix} 1 \\ 0 \\ 0 \\ 0 \end{bmatrix}$$

$$|01\rangle \equiv |0\rangle \otimes |1\rangle = \begin{bmatrix} 1 \\ 0 \end{bmatrix} \otimes \begin{bmatrix} 0 \\ 1 \end{bmatrix} = \begin{bmatrix} 0 \\ 1 \\ 0 \\ 0 \end{bmatrix}$$

$$|10\rangle \equiv |1\rangle \otimes |0\rangle = \begin{bmatrix} 0 \\ 1 \end{bmatrix} \otimes \begin{bmatrix} 1 \\ 0 \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \\ 1 \\ 0 \end{bmatrix}$$

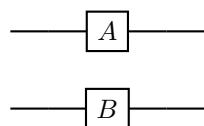
$$|11\rangle \equiv |1\rangle \otimes |1\rangle = \begin{bmatrix} 0 \\ 1 \end{bmatrix} \otimes \begin{bmatrix} 0 \\ 1 \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \\ 0 \\ 1 \end{bmatrix}$$

Given $|a\rangle = \alpha_0|0\rangle + \alpha_1|1\rangle$ and $|b\rangle = \beta_0|0\rangle + \beta_1|1\rangle$, we write $|a\rangle \otimes |b\rangle$ as

$$\begin{aligned} |a\rangle \otimes |b\rangle &= \begin{bmatrix} \alpha_0 \\ \alpha_1 \end{bmatrix} \otimes \begin{bmatrix} \beta_0 \\ \beta_1 \end{bmatrix} \\ &= \begin{bmatrix} \alpha_0 & \begin{bmatrix} \beta_0 \\ \beta_1 \end{bmatrix} \\ \alpha_1 & \begin{bmatrix} \beta_0 \\ \beta_1 \end{bmatrix} \end{bmatrix} \\ &= \begin{bmatrix} \alpha_0\beta_0 \\ \alpha_0\beta_1 \\ \alpha_1\beta_0 \\ \alpha_1\beta_1 \end{bmatrix}. \end{aligned}$$

Note that each element of the first vector multiplies the entire second vector. This is often the easiest way to get the tensor products in practice.

The matrix elements of the tensor product operation $A \otimes B$



are given by

$$(A \otimes B)_{ik,jl} = A_{ij}B_{kl}$$

where $ik \in \{00, 01, 10, 11\}$ labels the rows, and $kl \in \{00, 01, 10, 11\}$ labels columns, when forming the block matrix:

$$\begin{aligned} A \otimes B &= \begin{bmatrix} A_{00} & A_{01} \\ A_{10} & A_{11} \end{bmatrix} \otimes \begin{bmatrix} B_{00} & B_{01} \\ B_{10} & B_{11} \end{bmatrix} \\ &= \begin{bmatrix} A_{00}B & A_{01}B \\ A_{10}B & A_{11}B \end{bmatrix} \\ &= \left[\begin{array}{cc|cc} A_{00}B_{00} & A_{00}B_{01} & A_{01}B_{00} & A_{01}B_{01} \\ A_{00}B_{10} & A_{00}B_{11} & A_{01}B_{10} & A_{01}B_{11} \\ \hline A_{10}B_{00} & A_{10}B_{01} & A_{11}B_{00} & A_{11}B_{01} \\ A_{10}B_{10} & A_{10}B_{11} & A_{11}B_{10} & A_{11}B_{11} \end{array} \right] \end{aligned}$$

The Kronecker product.

This product $A \otimes B$ also known as the **Kronecker product** of matrices, which generalises the **outer product** of two vectors that we saw in Section 0.8.

The tensor product induces a natural partition of matrices into blocks. Multiplication of block matrices works pretty much the same as regular matrix multiplication (assuming the dimensions of the sub-matrices are appropriate), except that the entries are now matrices rather than numbers, and so may not commute.

- Evaluate the following matrix product of (4×4) block matrices:

$$\left[\begin{array}{c|c} 1 & X \\ \hline Y & Z \end{array} \right] \left[\begin{array}{c|c} 1 & Y \\ \hline X & Z \end{array} \right]$$

(where X , Y , and Z are the Pauli matrices).

- Using the block matrix form of $A \otimes B$ expressed in terms of A_{ij} and B_{ij} (as described above), explain how the following operations are performed on the block matrix:

- transposition $(A \otimes B)^T$;
- partial transpositions** $A^T \otimes B$ and $A \otimes B^T$;
- trace $\text{tr}(A \otimes B)$;
- partial traces** $(\text{tr } A) \otimes B$ and $A \otimes (\text{tr } B)$.

5.14.12 Hadamard transforms in components

Consider the Hadamard transform $H \otimes H \otimes H$ on three qubits, which is described by a $(2^3 \times 2^3)$ matrix. We know that

$$H = \frac{1}{\sqrt{2}} \begin{bmatrix} 1 & 1 \\ 1 & -1 \end{bmatrix}$$

and so we can calculate that

$$H \otimes H = \frac{1}{2} \begin{bmatrix} 1 & 1 & | & 1 & 1 \\ 1 & -1 & | & 1 & -1 \\ \hline 1 & 1 & | & -1 & -1 \\ 1 & -1 & | & -1 & 1 \end{bmatrix}$$

and thus that

$$H \otimes H \otimes H = \sqrt{\frac{1}{2^3}} \begin{bmatrix} 1 & 1 & | & 1 & 1 & | & 1 & 1 & | & 1 & 1 \\ 1 & -1 & | & 1 & -1 & | & 1 & -1 & | & 1 & -1 \\ \hline 1 & 1 & | & -1 & -1 & | & 1 & 1 & | & -1 & -1 \\ 1 & -1 & | & -1 & 1 & | & 1 & -1 & | & -1 & 1 \\ \hline 1 & 1 & | & 1 & 1 & | & -1 & -1 & | & -1 & -1 \\ 1 & -1 & | & 1 & -1 & | & -1 & 1 & | & -1 & 1 \\ \hline 1 & 1 & | & -1 & -1 & | & -1 & -1 & | & 1 & 1 \\ 1 & -1 & | & -1 & 1 & | & -1 & 1 & | & 1 & -1 \end{bmatrix}.$$

The rows and columns of $H \otimes H \otimes H$ are labelled by the triples $000, 001, \dots, 111$. Now, suppose we apply $H \otimes H \otimes H$ to the state $|110\rangle$:

$$\left. \begin{array}{l} |1\rangle \xrightarrow[H]{\frac{|0\rangle - |1\rangle}{\sqrt{2}}} \\ |1\rangle \xrightarrow[H]{\frac{|0\rangle - |1\rangle}{\sqrt{2}}} \\ |0\rangle \xrightarrow[H]{\frac{|0\rangle + |1\rangle}{\sqrt{2}}} \end{array} \right\} = \frac{1}{2^{3/2}} \begin{pmatrix} |000\rangle + |001\rangle - |010\rangle - |011\rangle \\ -|100\rangle - |101\rangle + |110\rangle + |111\rangle \end{pmatrix}$$

- The output state is a superposition of all binary strings: $\sum_{x \in \{0,1\}^3} c_x |x\rangle$. Where in the $H^{\otimes 3}$ matrix will you find the coefficients c_x ?

Do you want to write down $H \otimes H \otimes H \otimes H$? Probably not! This is an exponentially growing monster and you may soon run out of space if you actually do try to write it down. Instead, let us try to spot the pattern of the entries ± 1 in these matrices.

Consider again the single-qubit Hadamard gate matrix $H = (H_{ab})$, where $a, b = 0, 1$ are the labels for the rows and the columns. Observe that $H_{ab} = (-1)^{ab}/\sqrt{2}$. (This may look like a needlessly fancy way of writing the entries of the Hadamard matrix, but it will pay off in a moment).

- Using the fact that $(A \otimes B)_{ik,jl} = A_{ij}B_{kl}$, or any other method, analyse the pattern of the ± 1 in the tensor product of Hadamard matrices. What is the entry $H_{0101,1110}^{\otimes 4}$?
- For any two binary strings $a = (a_1, \dots, a_n)$ and $b = (b_1, \dots, b_n)$ of the same length we can define their “scalar” product as $a \cdot b = (a_1 b_1 \oplus \dots \oplus a_n b_n)$. Show that, up to the constant $(1/\sqrt{2})^n$, the entry $H_{a,b}^{\otimes n}$ is $(-1)^{a \cdot b}$ for any n and for any binary strings a and b of length n .
- Show that $H^{\otimes n}$ acts as

$$|a\rangle \mapsto \left(\frac{1}{\sqrt{2}}\right)^n \sum_{b \in \{0,1\}^n} (-1)^{a \cdot b} |b\rangle$$

- A quantum register of 10 qubits holds the binary string 0110101001. The Hadamard Transform is then applied to this register yielding a superposition of all binary strings of length 10. What is the sign in front of the $|0101010101\rangle$ term?

5.14.13 The Schmidt decomposition

An arbitrary vector in the Hilbert space $\mathcal{H}_A \otimes \mathcal{H}_B$ can be expanded in a product basis as

$$|\psi\rangle = \sum_{i,j} c_{ij} |a_i\rangle |b_j\rangle.$$

Moreover, for any given joint state $|\psi\rangle$, we can find orthonormal bases, $\{|\tilde{a}_i\rangle\}$ in \mathcal{H}_A and $\{|\tilde{b}_j\rangle\}$ in \mathcal{H}_B , such that $|\psi\rangle$ can be expressed as

$$|\psi\rangle = \sum_i d_i |\tilde{a}_i\rangle |\tilde{b}_i\rangle,$$

where the coefficients d_i are *non-negative* numbers. This is known as the **Schmidt decomposition** of $|\psi\rangle$.

Any bipartite state can be expressed in this form, but remember that *the bases used depend on the state being expanded*. Indeed, given two bipartite states $|\psi\rangle$ and $|\phi\rangle$, we usually *cannot* perform the Schmidt decomposition using the *same* orthonormal bases in \mathcal{H}_A and \mathcal{H}_B . The number of terms in the Schmidt decomposition is, at most, the minimum of $\dim \mathcal{H}_A$ and $\dim \mathcal{H}_B$.

The Schmidt decomposition follows from the **singular value decomposition** (often abbreviated to **SVD**): any $(n \times m)$ matrix C can be written as

$$C = UDV$$

where U and V are (respectively) $(n \times n)$ and $(m \times m)$ unitary matrices, and D is an $(n \times m)$ diagonal matrix with real, non-negative elements in descending order $d_1 \geq d_2 \geq \dots \geq d_{\min\{n,m\}}$ (and with the rest of the matrix is filled with zeros). The

elements d_k are called the **singular values** of C . We will return to the SVD in more detail later on, in Section 12.11.1.

You can visualize the SVD by thinking of C as representing a linear transformation from m -dimensional to n -dimensional Euclidean space: it maps the unit ball in the m -dimensional space to an ellipsoid in the n -dimensional space; the singular values are the lengths of the semi-axes of that ellipsoid; the matrices U and V carry information about the locations of those axes and the vectors in the first space which map into them. Thus SVD tells us that the transformation C consists of rotating the unit ball (the transformation V), stretching the k -th axis by a factor of d_k (the transformation D), and then rotating the resulting ellipsoid (the transformation U).

Using the index notation $C_{ij} = \sum_k U_{ik} d_k V_{kj}$, we can thus apply SVD to c_{ij} :

$$\begin{aligned} |\psi\rangle &= \sum_{i,j} c_{ij} |a_i b_j\rangle \\ &= \sum_{i,j} \sum_k U_{ik} d_k V_{kj} |a_i b_j\rangle \\ &= \sum_k d_k \left(\sum_i U_{ik} |a_i\rangle \right) \otimes \left(\sum_j V_{kj} |b_j\rangle \right). \end{aligned}$$

The Schmidt decomposition of a separable state of the form $|a\rangle \otimes |b\rangle$ is trivially just this state. The Bell states Ψ^+ and Φ^+ are already written in their Schmidt form, whereas Ψ^- and Φ^- can be easily expressed in the Schmidt form. For example, for $|\Psi^-\rangle$ we have $d_1 = d_2 = \frac{1}{\sqrt{2}}$, and the Schmidt basis is

$$\begin{aligned} |\tilde{a}_1\rangle &= |0\rangle \\ |\tilde{a}_2\rangle &= |1\rangle \\ |\tilde{b}_1\rangle &= |1\rangle \\ |\tilde{b}_2\rangle &= -|0\rangle. \end{aligned}$$

The number of non-zero singular values of c_{ij} is called the **rank** of c_{ij} , or the rank of the corresponding quantum state, or sometimes, the **Schmidt number**. You should be able to see that all bipartite states of rank-one are separable.

The Schmidt decomposition is *almost* unique. The ambiguity arises when we have two or more identical singular values, as, for example, in the case of the Bell states. Then any unitary transformation of the basis vectors corresponding to a degenerate singular value, both in \mathcal{H}_a and in \mathcal{H}_b , generates another set of basis vectors.

Part II

Further foundations

6 Bell's theorem

About quantum correlations, which are stronger than any correlations allowed by classical physics, and about the CHSH inequality (used to prove a variant of Bell's theorem) which demonstrates this fact.

Every now and then, it is nice to put down your lecture notes and go and see how things actually work in the real world. What is particularly wonderful (and maybe surprising) about quantum theory is that it turns up in many places where we might not expect it to. One such example is in the polarisation of light, where we stumble across an intriguing paradox.

The (much-simplified) one sentence introduction to light polarisation is this: light is made of **transverse** waves, and transverse waves have a “direction”, which we call **polarisation**; a **polarising filter** only allows waves of a certain polarisation to pass through. If we take two polarising filters, and place them on top of each other with their polarisations oriented at 90° to one another, then basically no light will pass through, since the only light that can pass through the first filter is orthogonally polarised with respect to the second filter, and is thus blocked from passing through. But then, if we take a third filter, and place it in between the other two, at an angle in the middle of both (i.e. at 45°), then somehow *more* light is let through than if the middle filter weren't there at all.

This is intrinsically linked to **Bell's theorem**, which proves the technical sounding statement that “any local real hidden variable theory must satisfy certain statistical properties”, which is *not* satisfied in reality, as many quantum mechanical experiments (such as the above) show!

For the more visually inclined, there is a [video on YouTube by minutephysics](#) about this experiment, or you can play with a virtual version on the [Quantum Flytrap Virtual Lab](#).

6.1 Hidden variables

The story of “hidden variables” dates back to 1935 and grew out of Einstein’s worries about the completeness of quantum theory. Consider, for example, a single qubit. Recalling our previous discussion on compatible operators (Section 4.6), we know that no quantum state of a qubit can be a simultaneous eigenstate of two *non-commuting* operators, such as σ_x and σ_z . Physically, this means that if the qubit has a definite value of σ_x then its value of σ_z must be indeterminate, and vice versa. If we take quantum theory to be a complete description of the world, then we must accept that it is impossible for both σ_x and σ_z to have definite values for the same qubit at the same time. Einstein felt very uncomfortable about all this: he argued that quantum theory is incomplete, and that observables σ_x and σ_z may both have simultaneous definite values, although we only have knowledge of one of them at a time. This is the hypothesis of **hidden variables**.

Here it's important that we're really talking about so-called **local** hidden variable theories. We discuss the technical details in 6.7.

In this view, the indeterminacy found in quantum theory is merely due to our ignorance of these “hidden variables” that are present in nature but not accounted for in the theory. Einstein came up with a number of pretty good arguments for the existence of “hidden variables”, perhaps the most compelling of which was described in his 1935 paper (known as “the EPR paper”), co-authored with his younger colleagues, Boris Podolsky and Nathan Rosen. It stood for almost three decades as the most significant challenge to the completeness of quantum theory. Then, in 1964, John Bell showed that the local hidden variable hypothesis can be tested and *refuted*.

This key word “local” is very important for those who care about the subtle technical details, but we won’t explain it here.

Any theory can make predictions, but just because the predictions turn out to be correct, this does not make the theory true — there may be other, maybe equivalent, explanations. The key to the scientific method is **falsifiability**: make one prediction incorrectly, and you have proven your theory is not true.

Hidden-variable no-go theorems.

We already saw some no-go theorems in Section 5.9 that set limits on what we can do with quantum states. In this chapter we're going to see one no-go theorem relating to the *foundations* of quantum theory, in particular concerning these *local* “hidden variables”. Again, there are many related no-go theorems, and again they fall beyond the scope of this book, but it's worth mentioning them by name at least. They all state that a certain type of (**realistic**, in some technical sense of the word) hidden-variable theory is inconsistent with reality:

- **Bell's theorem** (which we will see in Section 6.4) is for **local** hidden-variable theories.
- The **Kochen–Specker theorem** is for **non-contextual** hidden-variable theories.
- The **Pusey–Barret–Rudolph theorem** (often simply called the **PBR theorem**) is for **preparation independent** hidden-variable theories.

All together, these three theorems say that, if some hidden-variable theory does exist, then it has to be *non-local*, *contextual*, and *preparation dependent*. But what do these words mean?

Preparation independence is the assumption that, if we independently prepare two quantum states, then their hidden variables are also independent. Locality is the idea that things can only be directly affected by their surroundings, i.e. the exact opposite of “spooky action at a distance”. Contextuality is a bit more subtle, and can actually be seen as a direct generalisation of non-locality (by **Fine's theorem**), but it talks about how results of measurements depend on the commutator of the observable being measured, i.e. on its “context”.

A particularly useful way of formally defining non-locality and contextuality is by using the language of **sheaf theory**, which is an inherently topological and category-theoretic notion. This approach was cemented by Abramsky and Brandenburger’s “The Sheaf-Theoretic Structure Of Non-Locality and Contextuality”, arXiv:[1102.0264](https://arxiv.org/abs/1102.0264).

6.2 Quantum correlations

Consider two entangled qubits in the **singlet** state

$$|\psi\rangle = \frac{1}{\sqrt{2}} (|01\rangle - |10\rangle)$$

and note that the projector $|\psi\rangle\langle\psi|$ can be written as

$$|\psi\rangle\langle\psi| = \frac{1}{4} (\mathbf{1} \otimes \mathbf{1} - \sigma_x \otimes \sigma_x - \sigma_y \otimes \sigma_y - \sigma_z \otimes \sigma_z)$$

where σ_x , σ_y , and σ_z are our old friends the Pauli matrices.

Also recall that any single-qubit observable with values ± 1 can be represented by the operator

$$\vec{a} \cdot \vec{\sigma} = a_x \sigma_x + a_y \sigma_y + a_z \sigma_z,$$

We say that a system is **singlet** if all the qubits involved are entangled. For example, the Bell states (Section 5.7) are all (maximally entangled) singlet states. This is related to the notion of **singlet states** in quantum mechanics, which are those with zero net angular momentum.

We say “observable” and “value” instead of “Hermitian operator” and “eigenvalue” because it's useful to be able to switch between speaking like a mathematician and like a physicist!

where \vec{a} is a unit vector in the three-dimensional Euclidean space.

So if Alice and Bob both choose observables, then we can characterise their choice by vectors \vec{a} and \vec{b} , respectively. If Alice measures the first qubit in our singlet state $|\psi\rangle$, and Bob the second, then the corresponding observable is described by the tensor product

$$A \otimes B = (\vec{a} \cdot \vec{\sigma}) \otimes (\vec{b} \cdot \vec{\sigma}).$$

The eigenvalues of $A \otimes B$ are the products of eigenvalues of A and B . Thus $A \otimes B$ has two eigenvalues: +1, corresponding to the instances when Alice and Bob registered identical outcomes, i.e. $(+1, +1)$ or $(-1, -1)$; and -1, corresponding to the instances when Alice and Bob registered different outcomes, i.e. $(+1, -1)$ or $(-1, +1)$.

This means that the expected value of $A \otimes B$, in any state, has a simple interpretation:

$$\langle A \otimes B \rangle = \Pr(\text{outcomes are the same}) - \Pr(\text{outcomes are different}).$$

This expression can take any real value in the interval $[-1, 1]$, where -1 means we have **perfect anti-correlations**, 0 means **no correlations**, and +1 means **perfect correlations**.

We can evaluate the expectation value in the singlet state:

$$\begin{aligned} \langle \psi | A \otimes B | \psi \rangle &= \text{tr} [(\vec{a} \cdot \vec{\sigma}) \otimes (\vec{b} \cdot \vec{\sigma}) | \psi \rangle \langle \psi |] \\ &= -\frac{1}{4} \text{tr} [(\vec{a} \cdot \vec{\sigma}) \sigma_x \otimes (\vec{b} \cdot \vec{\sigma}) \sigma_x + (\vec{a} \cdot \vec{\sigma}) \sigma_y \otimes (\vec{b} \cdot \vec{\sigma}) \sigma_y + (\vec{a} \cdot \vec{\sigma}) \sigma_z \otimes (\vec{b} \cdot \vec{\sigma}) \sigma_z] \\ &= -\frac{1}{4} \text{tr} [4(a_x b_x + a_y b_y + a_z b_z) \mathbf{1} \otimes \mathbf{1}] \\ &= -\vec{a} \cdot \vec{b} \end{aligned}$$

where we have used the fact that $\text{tr}(\vec{a} \cdot \vec{\sigma})\sigma_k = 2a_k$ (for $k = x, y, z$). So if Alice and Bob choose the same observable $\vec{a} = \vec{b}$, then the expected value $\langle A \otimes B \rangle$ will be equal to -1, and their outcomes will *always* be opposite: whenever Alice registers +1 (resp. -1) Bob is bound to register -1 (resp. +1).

6.3 The CHSH inequality

An upper bound on classical correlations.

We will describe the most popular version of Bell's argument, introduced in 1969 by [John Clauser](#), [Michael Horne](#), [Abner Shimony](#), and [Richard Holt](#) (whence the name "CHSH").

Let us start by making this assumption that the results of any measurement on any individual system are predetermined — any probabilities we may use to describe the system merely reflect our ignorance of these hidden variables.

Imagine the following scenario. Alice and Bob, our two characters with a predilection for wacky experiments, are equipped with appropriate measuring devices and sent to two distant locations. Assume that Alice and Bob each have a choice of *two* observables that they can measure, each with well defined values +1 and -1. Let's say that Alice can choose between observables A_1 and A_2 , and Bob between B_1 and B_2 . Now, somewhere in between them there is a source that emits pairs of qubits that fly apart, one towards Alice and one towards Bob. For each incoming qubit, Alice and Bob choose randomly, and independently from each other, which particular observable will be measured. This means we can think of the observables as random variables A_k, B_k (for $k = 1, 2$) that take values ± 1 . Using these, we can define a new random variable: the **CHSH quantity**

$$S = A_1(B_1 - B_2) + A_2(B_1 + B_2).$$

For example, if the two qubits are spin-half particles, they may measure the spin components along the directions \vec{a} and \vec{b} .

Recall Section 4.5: the expected value of an operator E in the state $|\phi\rangle$ is equal to $\langle \phi | E | \phi \rangle$.

The phrase "well defined" corresponds to our "hidden variable" assumption, i.e. that the observables *always* have *definite* values.

By a case-by-case analysis of the four possible outcomes for the pair (B_1, B_2) , we see that one of the terms $B_1 \pm B_2$ must be equal to zero and the other to ± 2 (basically depending on if $B_1 = B_2$ or not), and so (looking at the four possible outcomes for the pair (A_1, A_2)) we see that $S = \pm 2$. But the average value of S must lie in between these two possible outcomes, i.e.

$$-2 \leq \langle S \rangle \leq 2.$$

That's it! Such a simple and yet profound mathematical statement about correlations, which we refer simply to as the **CHSH inequality**.

There is absolutely *no quantum theory involved* in the CHSH inequality

$$-2 \leq \langle S \rangle \leq 2$$

because the CHSH inequality is not specific to quantum theory: it does not really matter what kind of physical process is behind the appearance of binary values of A_1 , A_2 , B_1 , and B_2 ; it is merely a statement about correlations, and for all classical correlations we must have

$$|\langle A_1 B_1 \rangle - \langle A_1 B_2 \rangle + \langle A_2 B_1 \rangle + \langle A_2 B_2 \rangle| \leq 2$$

(which is just another way of phrasing the CHSH inequality).

There are essentially *two* (very important) assumptions here:

1. **Hidden variables.** Observables have definite values.
2. **Locality.** Alice's choice of measurements (choosing between A_1 and A_2) does not affect the outcomes of Bob's measurement, and vice versa.

We will not discuss the locality assumption right now in detail (see Section 6.7), but let us just give one brief comment. In the hidden variable world a, statement such as "if Bob were to measure B_1 then he would register $+1$ " must be either true or false (and not "undecidable" or some other such thing!) *prior to Bob's measurement*. Without the locality hypothesis, such a statement is ambiguous, since the value of B_1 could depend on whether A_1 or A_2 will be chosen by Alice. We do not want this since it implies *instantaneous communication* — it means that, say, Alice by making a choice between A_1 and A_2 affects Bob's results: Bob can immediately "see" what Alice "does".

Now let's see how quantum theory *fundamentally disagrees* with the CHSH inequality.

6.4 Bell's theorem via CHSH

Continuing this story of Alice and Bob with their observables and pairs of qubits, let us first rephrase things in the formalism of quantum mechanics that we've been building up. The observables A_1 , A_2 , B_1 , B_2 become (2×2) Hermitian matrices, each with the two eigenvalues ± 1 , and $\langle S \rangle$ becomes the expected value of the (4×4) **CHSH matrix**

$$S = A_1 \otimes (B_1 - B_2) + A_2 \otimes (B_1 + B_2).$$

We can now evaluate $\langle S \rangle$ using quantum theory.

Actually performing these measurements described by S on a pair of qubits is known as a **CHSH test**, or **Bell test**.

If the two qubits are in the singlet state

$$|\psi\rangle = \frac{1}{\sqrt{2}}(|01\rangle - |10\rangle)$$

then we have already seen (Section 6.2) that

$$\langle A \otimes B \rangle = -\vec{a} \cdot \vec{b}.$$

So if we choose vectors $\vec{a}_1, \vec{a}_2, \vec{b}_1$, and \vec{b}_2 as shown in Figure 6.1, then the corresponding matrices satisfy

$$\begin{aligned}\langle A_1 \otimes B_1 \rangle &= \langle A_2 \otimes B_1 \rangle = \langle A_2 \otimes B_2 \rangle = \frac{1}{\sqrt{2}} \\ \langle A_1 \otimes B_2 \rangle &= -\frac{1}{\sqrt{2}}.\end{aligned}$$

That is, $A_1 = \vec{a}_1 \cdot \vec{\sigma}$, and so on.

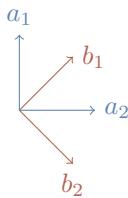


Figure 6.1: The relative angle between the two perpendicular pairs is 45° .

Plugging these values in, we get that

$$\langle A_1 B_1 \rangle - \langle A_1 B_2 \rangle + \langle A_2 B_1 \rangle + \langle A_2 B_2 \rangle = -2\sqrt{2},$$

which obviously violates CHSH inequality: $-2\sqrt{2}$ is strictly less than -2 !

But here is the really important part of this discussion: *this violation of the CHSH has been observed in a number of painstakingly careful experiments — this is not just theoretical!* The early efforts in these experiments were truly heroic, with many many layers of complexity; today, however, such experiments are routine.

Bell's theorem. The behaviour of entangled quantum systems *cannot* be explained by local hidden variables. In other words, outcomes in quantum mechanics really are random, and it's not simply our lack of knowledge about some background process.

If we can enforce locality in an experimental setup (for example, by ensuring that Alice and Bob are sufficiently far apart so that there is not enough time between Alice making a measurement and Bob receiving his measurement result) then an experimental verification of the CHSH test proves to us that the system is behaving in an inherently non-classical and, importantly, *unpredictable* manner. This means that this is a good test to see if our devices are performing as they are supposed to, and are untampered by any potential eavesdroppers. In other words, the CHSH test is key for securing quantum protocols, as we will explain in Section 6.6.

6.5 Tsirelson's inequality

An upper bound on quantum correlations.

If an eavesdropper has observed our system to the extent that they can predict our outcomes, then that very predictability means that there is a hidden-variable description of the system, and the CHSH inequality is not violated.

One may ask if $|\langle S \rangle| = 2\sqrt{2}$ is the *maximal* violation of the CHSH inequality, and the answer is “yes, it is”: *quantum* correlations *always* satisfy the bound $|\langle S \rangle| \leq 2\sqrt{2}$. This is because, no matter which state $|\psi\rangle$ we pick, the expected value $\langle S \rangle = \langle \psi | S | \psi \rangle$ cannot exceed the largest eigenvalue of S , and we can put an upper bound on the largest eigenvalues of S . To start with, taking the largest eigenvalue (in absolute value) of a Hermitian matrix M , which we denote by $\|M\|$, gives a matrix norm, i.e. it has the following properties:

$$\begin{aligned}\|M \otimes N\| &= \|M\| \|N\| \\ \|MN\| &\leq \|M\| \|N\| \\ \|M + N\| &\leq \|M\| + \|N\|\end{aligned}$$

Given that $\|A_k\| = \|B_k\| = 1$ (for $k = 1, 2$), it is easy to use these properties to show that $\|S\| \leq 4$, but this is a much weaker bound than we want. However, one can show that

$$S^2 = 4(\mathbf{1} \otimes \mathbf{1}) + [A_1, A_2] \otimes [B_1, B_2].$$

Now, the norms of the commutators $\|[A_1, A_2]\|$ and $\|[B_1, B_2]\|$ are bounded by 2, and $\|S^2\| = \|S\|^2$. All together, this gives

$$\begin{aligned}\|S^2\| &\leq 8 \\ \implies \|S\| &\leq 2\sqrt{2} \\ \implies |\langle S \rangle| &\leq 2\sqrt{2}\end{aligned}$$

This result is known as the **Tsirelson inequality**.

In *classical* probability theory, the (absolute value of the) average value of the CHSH quantity

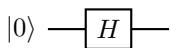
$$S = A_1(B_1 - B_2) + A_2(B_1 + B_2)$$

is bounded by 2, and this bound can be attained.

In *quantum* theory, the same value is bounded by $2\sqrt{2}$, and this bound can also be attained.

6.6 Quantum randomness

The experimental violations of the CHSH inequality have shown us that there are situations in which the measurement outcomes are truly unknown the instant before the measurement is made, and so the answer must be “chosen” randomly. We can make use of this randomness in a number of different ways, the most obvious example of which being a random number generator. Indeed, we have already met one suitable implementation:



The state before measurement is $(|0\rangle + |1\rangle)/\sqrt{2}$, so the two possible outcomes occur with equal probability. This is a truly random number generator, not like the pseudorandom one that is used if you ask your computer for some random data.

This randomness generator works well as long as we know how it's been built, i.e. that it really is just a Hadamard gate, that the input qubit really has been prepared in the state $|0\rangle$, and that the measurement device is accurate and honest. However, we don't all have a Hadamard gate and a supply of prepared qubits lying around at

home, so it seems likely that at some point we might have to buy or borrow such a device from a third party. But then how can we *know* that it really is doing what it promises, and not just supplying some pseudorandom numbers that might, for example, already be known to the manufacturer? This would render the device useless for cryptographic purposes! But not only do we want to know that this isn't the case, we would also like the average user to be able to verify this for themselves, without having to know about the internal details. In other words, can we find a way of verifying the device via some analysis of just inputs and outputs? This is the question of device independence.

A protocol is **device independent** if its security doesn't depend on trusting the devices on which it is implemented. In other words, it has no reliance on trusting the third party who supply you with the devices.

We can rule out one thing from the start, namely deterministic behaviour. If we behave deterministically then we have no hope, since the third party can take this into account and potentially find a way to always fool us. But there is another approach that we can try: rather than directly trying to verify the veracity of any given device, we can try to use it to turn a small amount of true randomness into a larger amount. This is the idea of randomness expansion.

Starting from an initial seed of private randomness (completely unknown to any other party), **randomness expansion** is the process of extending this to a larger amount of randomness that remains completely private.

Let's consider a different device: one that produces pairs of qubits in singlet states and gives one of its qubits to Alice and one to Bob. If Alice then measures her qubit in the X basis, and Bob measures his in the Z basis (each keeping their outcomes private), they each obtain random bits that are independent of one another. However, this is only really true if the device truly is giving them singlet states, and not predetermined unentangled states. How can they test for this?

Using the idea of randomness expansion, let's assume that they start with some shared random private seed: some m -bit string that only they know. They start by generating n of these putative singlet states, and publicly decide on some value $0 < p < 1$. With this, they randomly select $[pn]$ of the pairs to perform a CHSH test on. Each test requires two random bits (to determine Alice and Bob's choice of measurement), so in total we will need the length m of their shared random private seed be roughly

$$m \approx 2pn - pn \log_2 p - n(1-p) \log_2(1-p)$$

where the log terms are approximately how many bits are required to randomly choose the subset of pairs to test.

Why does this help? Well, if somebody has manipulated the device that produces the pairs, then they need to be sure that they haven't altered the pairs that Alice and Bob are testing on. But they cannot know in advance which pairs that will be, and so they cannot risk manipulating anything.

In the fraction p of pairs where a CHSH test is performed, we get at least 1 bit of randomness out: Alice's measurement result is always chosen at random. In fact, Bob's result will be partially correlated to Alice's, so we should be able to extract some more randomness from this as well, but we ignore this possibility for the sake of simplicity. Thus in the end we create $(2-p)n$ bits of randomness.

Note that we're pushing the problem somewhere else: how can they come up with this shared private seed in the first place? This is the problem of **key distribution**, and we'll return to this again later.

One can be much more quantitative about this by using Chernoff bounds for a simple strategy of "choose at random which pairs to manipulate", but a full proof of security is much more involved than we would like to be here.

6.7 Loopholes in Bell tests

When we introduced the idea of hidden variable theories in Section 6.1, we made some assumptions to simplify the exposition, but these have a big impact on the practical reality of violating Bell tests. Any test that does not satisfy one or more of these assumptions is said to have a **loophole**. For verifying fundamental physics, we are not so worried about these loopholes — it feels very unlikely that the putative classicality of the experiment is hiding in whatever loophole might be available in a given system. But for cryptographic purposes, an adversary will use and loophole at their disposal to try to trick you!

There are three types of key assumptions that we will talk about here, and for each one we provide some exercises to work through in order to explore them further:

- detector efficiency (Exercise 6.8.7)
- locality (Exercise 6.8.8)
- free will (Exercise 6.8.9).

Let's start with the first, which gives rise to the **detector loophole**. When we make a measurement with a real-life device, in practice it doesn't always work — maybe it just fails to notice a photon flying past. Each detector has a parameter η known as its **efficiency**: η is the probability that the measurement succeeds. For testing fundamental physics, it seems reasonable to assume that the successful measurements are a fair sample of what's really going on. But if there's an adversary, they might substitute our detectors for completely perfect one, and then deliberately choose to fake a failure whenever their eavesdropping attempts fail.

The next crucial assumption in the CHSH test is that Alice and Bob are separated by a “large enough” distance in space and time. More precisely, if they are physical at distance L from each other, then their random choices of measurement setting, followed by their corresponding carrying out of the measurement, and receipt of the answers, should all be accomplished within a time approximately L/c of each other, where c is the speed of light. If Alice and Bob are not far enough away from each other, then they are said to be within each other's **locality**, and so this is known as the **locality loophole**.

The final important assumption that we will mention here involves the availability of true randomness, and emphasises the importance of randomness expansion. It asserts that Alice and Bob must be able to choose their measurement settings randomly. This freedom to make their own choices is glibly referred to as them having “free will”, and so this is known as the **free-will loophole**. Resolving the locality loophole puts extremely tight constraints on how quickly choices must be made, to the extent that Alice and Bob cannot make those choices manually — they need to use random number generators. But then they need to be able to trust that these generators are indeed random, not merely pseudorandom, otherwise somebody else could know the origin of the “random” numbers and use that information to their advantage.

We say “approximately” here because we are avoiding being specific about how we actually define distance.

If we say that Alice and Bob are $L = 30$ km apart from each other, then we're talking of timescales on the order of 10^{-4} s, which is not very long!

6.8 Remarks and exercises

6.8.1 XOR games

The setup of the CHSH inequality that we have described can instead be imagined as a two-player all-or-nothing game between Alice and Bob, so let's study this type of game more generally.

- Alice and Bob each start with an integer **prompt** a and b (respectively), with $1 \leq a, b \leq n$. This integer can come from anywhere: they could pick it themselves, or it could be given to them. Having seen this integer, a round of the game consists of them returning an **answer**, x and y (respectively), of length m bits to an oracle. Both the prompt and the answer are kept secret, so that only the corresponding player knows them.

- They win if the **winning condition** computed by the oracle is 1, and lose if it is 0. The winning condition is given by

$$\begin{cases} 1 & \text{if } g_A(x, a, b) = g_B(y, a, b) \\ 0 & \text{if } g_A(x, a, b) \neq g_B(y, a, b) \end{cases}$$

where g_A and g_B are deterministic one-bit-valued functions whose output values are equally likely to be 0 or 1 (i.e. they return the value 0 for exactly half of the possible input triples).

- There exists a *quantum* strategy that always wins. It uses sets of m measurements on a maximally entangled state, and the measurement operators for all possible settings either commute or anticommute with one another. The measurements are specified by observables that are traceless (i.e. their trace is equal to 0) and square to the identity.
- The best possible classical strategy fails f of the time, for some fraction f .

6.8.2 XOR games for quantum key distribution

We can use the setup from Exercise 6.8.1 as the basis for a **quantum key distribution scheme**: a process that allows two people to produce a shared random string known only to them.

For the i -th round of the game, Alice and Bob randomly (and privately) select their prompts a_i and b_i , which they then use to play one round of the game, giving answers x_i and y_i . They keep a note of all their prompts and answers, as well as the result of the winning condition for each round. After many rounds, Alice and Bob publicly announce all their prompts $\{a_1, \dots, a_n\}$ and $\{b_1, \dots, b_n\}$.

This means that Alice, for example, now knows the following:

- both sets of prompts $\{a_1, \dots, a_n\}$ and $\{b_1, \dots, b_n\}$
- her answers $\{x_1, \dots, x_n\}$
- the values of $i \in \{1, \dots, n\}$ for which $g_A(x_i, a_i, b_i) = g_B(y_i, a_i, b_i)$, i.e. the numbers i_1, \dots, i_k of the rounds that they won.

Bob knows the same, but instead of Alice's answers $\{x_1, \dots, x_n\}$ he knows his own $\{y_1, \dots, y_n\}$.

This is enough information (given that we have run enough rounds) for Alice to determine g_A and Bob to determine g_B . The two of them can then each compute the k -bit string

$$g_A(x_{i_1}, a_{i_1}, b_{i_1}) \dots g_A(x_{i_k}, a_{i_k}, b_{i_k}) = g_B(x_{i_1}, a_{i_1}, b_{i_1}) \dots g_B(x_{i_k}, a_{i_k}, b_{i_k})$$

which gives their shared key. Now they just need to deal with eavesdropping.

By publicly selecting a random selection of rounds and announcing the results of their putative g_A and g_B for these rounds, they can check whether or not their values are coherent with the result of the winning conditions determined by the oracle.

- If everything is coherent with the results, then they can be sure (if they have played enough rounds and compared enough results) that there was no eavesdropping.
- If the fraction of tests that are coherent with the results is less than f , then they must assume that somebody has been eavesdropping, and they should cease communication.
- Anywhere in between these two cases, they can quantify how much an eavesdropper might know, and then run some method of privacy amplification to further exclude the eavesdropper (at the cost of shortening the key).

For example, if they know that they lost round i , then it should be the case that $g_A(x_i, a_i, b_i) \neq g_B(y_i, a_i, b_i)$.

6.8.3 XOR games for randomness expansion

Consider again the scenario of Exercise 6.8.1. Explain how Alice by herself can treat this as a single-player game and use it as the basis for a randomness expansion scheme.

Hint: unlike in Exercise 6.8.2, Alice no longer needs to choose a random subset of rounds to check the winning condition for: she can check all of them.

6.8.4 Prescribed binary randomness

Find a quantum circuit that can act as a random source that outputs 0 with probability $(2 + \sqrt{2})/4$ and 1 with probability $(2 - \sqrt{2})/4$. *Hint: what is $\cos^2(\pi/4)$?*

6.8.5 Unbiasing bias

Suppose you have a source that produces random bits, but operates with a bias: it outputs 0 with probability p and 1 with probability $1 - p$, for some fixed $0 < p < 1$.

Find a method such that, given two outputs from this source, you successfully obtain a single unbiased random bit (i.e. as if the source had $p = 1/2$) with probability $2p(1 - p)$.

6.8.6 Proving Tsirelson's inequality

Let A_i , B_i , and $\| - \|$ be as in Section 6.5.

1. Prove that

$$(A_1 \otimes (B_1 - B_2) + A_2 \otimes (B_1 + B_2))^2 = 4(\mathbf{1} \otimes \mathbf{1}) + [A_1, A_2] \otimes [B_1, B_2].$$

2. Prove that

$$\|[A_1, A_2]\| \leq 2$$

(and the same argument should also apply to $\|[B_1, B_2]\|$).

6.8.7 Detector loophole

Say we have a detector with efficiency η , and an otherwise perfect CHSH test with $\langle S \rangle = 2\sqrt{2}$.

1. With what probability do both detections succeed? With what probability does exactly one detection fail? With what probability do both detectors fail?
2. Imagine that one detector successfully measures a qubit of a Bell state, while the other detector fails and notifies us of this fact. If we replace the reading of the failed detector by +1, what is the average value of the outcome?
3. Now imagine that both detections fail, and both readings are replaced by +1. What is the average value of the outcome?
4. Using the above, show that the critical detector efficiency for being able to rely on the outcome of the CHSH test is given by

$$\eta = 2(\sqrt{2} - 1).$$

Hint: if both detections succeed, then we can achieve $\langle S \rangle = 2\sqrt{2}$; the previous questions calculate $\langle S \rangle$ in the other possible cases; so what is the sum of these values, weighted by their probabilities of occurring?

6.8.8 Locality loophole

1. Imagine that Alice and Bob are not very far from each other, and that they are going to perform a CHSH test using devices given to them by Eve, who has tampered with the devices and knows *both* of the measurement settings. How can Eve make Alice and Bob believe that they are sharing Bell pairs when, in reality, they are not?

To really get into the details of locality, we need to use some tools from [special relativity](#): the theory of how non-accelerating observers measure times and distances. The main assertion of special relativity is that *nothing can travel faster than the speed of light*. These next exercises will be much easier if you have already seen things such as Minkowski diagrams before, but do not worry if you haven't.

2. We are going to plot a **Minkowski diagram** of the CHSH test scenario. This is a plot of physical position along the horizontal axis against time on the vertical axis. We place one **event** — let's pick Alice choosing her measurement setting and getting a measurement result — at the origin. Include on this diagram all the “places” (a pair consisting of a space coordinate and a time coordinate) that can receive a message about what measurement setting Alice chose, appealing to the main assertion of special relativity. This set of places is called the **future** of the event.
3. Add to the Minkowski diagram all the places that can send a message that could influence Alice’s outcomes. This set of places is called the **past** of the event.
4. If an event (such as Bob choosing a measurement setting and getting a result) occurs in a region that is in neither the future nor the past of the event at the origin, what influence can these two events have over one another? If Bob’s event is at a distance L along the x -axis from the origin, then what is the maximal permissible time difference between the two events?
5. If we wish to be really careful, then we should separate out the four events:
 - a. Alice chooses a measurement setting
 - b. Alice gets a measurement result
 - c. Bob chooses a measurement setting
 - d. Bob gets a measurement result.

Draw a Minkowski diagram that includes all four events. Assuming that Alice and Bob are stationary, use this diagram to more explicitly describe the timing constraints of when each event should happen relative to one another if we wish to avoid the locality loophole.

To make things easy, we assume that space is one-dimensional: Alice and Bob live on a line.

Hint: we already said in Section 6.7 that the maximal permissible time difference should be L/c , so prove this.

6.8.9 Free-will loophole

1. Assume that Eve, the manufacturer of the devices performing the CHSH test, knows what settings Alice and Bob are going to use. Explain how Eve can have faked the outputs, making the predetermined, while still seeming to be producing results consistent with quantum violation of the CHSH inequality.
2. Say that Eve only knows a fraction p of the random outcomes (separately for both Alice and Bob). What is the maximum value of p that still allows $\langle S \rangle = 2\sqrt{2}$? What about merely allowing $\langle S \rangle > 2$? (Assume that, whenever at least one random number is known, both devices know that value, and also know that they don’t know the other value, but that one device will learn it when Alice or Bob chooses the measurement setting).
3. Imagine that Alice has a string of length k of (apparently) randomly chosen bits. She believes that Eve knows a fraction p of these bits. In an attempt to thwart Eve’s attempts at eavesdropping, Alice computes the addition modulo 2 of all k bits, resulting in a single bit. What is the probability that Eve can know this final value?

Depending on how you answered Exercise 6.8.8, you might be able to use exactly the same idea here.

7 Stabilisers

About the structure of the **Pauli group**, which is the group generated by tensor products of the Pauli matrices, including the identity. It has nice algebraic properties which are useful in many areas of quantum information science, in particular quantum error correction and classical simulations of some types of quantum computation. We will discuss how certain subgroups of the Pauli group, and in particular **stabilisers** and **normalisers** of these subgroups, slice the Pauli group into interesting **cosets** that have a group structure of their own. We will also look at the **Clifford group**, which is a set of unitary operators that preserve the Pauli group under conjugation and describes the “easy” part of quantum computation.

N.B. This section is sort of an odd-one-out, since we won’t need any of this formalism until Sections 13 and 14. However, if you’re reading this book in order, then you might find this a nice detour halfway through, and it gives a taste of things to come.

We have already seen the (single-qubit) Pauli matrices, along with a brief look into their algebraic structure, in Section 3.3.

$$\mathbf{1} = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix} \quad X = \begin{bmatrix} 0 & 1 \\ 1 & 0 \end{bmatrix} \quad Y = \begin{bmatrix} 0 & -i \\ i & 0 \end{bmatrix} \quad Z = \begin{bmatrix} 1 & 0 \\ 0 & -1 \end{bmatrix}$$

Recall that these matrices span the entire space of (2×2) complex matrices, square to the identity (and thus can only have eigenvalues in the set $\{+1, -1\}$), and are both Hermitian and unitary. As such, they can represent both observables *and* unitary evolutions. Any two given Pauli matrices either commute or anticommute.

As one final reminder, we often refer to the Pauli matrices as “matrices”, but they are defined as operators by the commutations relations, without reference to any particular basis. That is, the Pauli operators X , Y , and Z are defined exactly by the relations

$$\begin{aligned} X^2 &= Y^2 = Z^2 = \mathbf{1} \\ XY &= iZ \quad YZ = iX \quad ZX = iY \\ YX &= -iZ \quad ZY = -iX \quad XZ = -iY. \end{aligned}$$

7.1 Pauli groups

When we multiply the four Pauli matrices with one another we get Pauli matrices in return, but with possible phase factors ± 1 and $\pm i$ (e.g. $XY = iZ$). Once we include these phase factors, ensuring that we have a set that is closed under matrix multiplication, we obtain the **single qubit Pauli group**, which we denote by \mathcal{P}_1 .

In order to characterise a group, we can simply list all its elements and define the group operation on each possible pair, but it is usually more efficient to use the notion of **group generators**. Given a group G , these are elements g_1, \dots, g_n of the group that are **independent** (we cannot write any one of them as a product of some of the others) and such that every element of G can be written as a product of (possibly repeated) elements of $\{g_1, \dots, g_n\}$. If G is generated by g_1, \dots, g_n , then we write $G = \langle g_1, \dots, g_n \rangle$.

Note how similar this is to the definition of a basis for a vector space.

The **single-qubit Pauli group** \mathcal{P}_1 is defined by

$$\begin{aligned}\mathcal{P}_1 &:= \langle X, Y, Z \rangle \\ &= \{\pm 1, \pm i\mathbf{1}, \pm X, \pm iX, \pm Y, \pm iY, \pm Z, \pm iZ\}.\end{aligned}$$

The **n -qubit Pauli group** \mathcal{P}_n is defined to consist of all n -fold tensor products of Pauli matrices, with possible global phase factors ± 1 and $\pm i$, i.e.

$$\mathcal{P}_n := \{P_1 \otimes \dots \otimes P_n \mid P_1, \dots, P_n \in \mathcal{P}_1\}.$$

This group has 4^{n+1} elements: 4×4^n , since we have to account for the possible global phase factors (which usually aren't very important for practical applications, but are necessary in order to have a well defined group).

As a small mathematical aside, we could use some group theory here: \mathcal{P}_n has two trivial (multiplicative) subgroups, namely $Z_2 = \{\pm 1\}$ and $Z_4 = \{\pm 1, \pm i\}$; the quotient group \mathcal{P}_n/Z_4 is exactly the n -qubit Pauli group but with the phases ignored.

Some researchers prefer to think of the (single-qubit) Pauli group as the group generated only by X and Z (leaving out Y), which then only has 8 elements: $\pm 1, \pm X, \pm Z$, and $\pm iY$. We do **not** follow this convention.

Central products.

One abstract way of defining the Pauli group, without having to make any reference to matrices (and thus to bases), or even to operators, is using the notion of a **central product**. This is a way of combining two smaller group into one large group, but “respecting” the commutative parts of each, which means that it arises as a quotient of the **direct product** (which is somehow the most blunt way of combining together two groups).

The **cyclic group of order 4** is the abstract manifestation of something maybe more familiar: the additive group of integers modulo 4. That is, the numbers 0, 1, 2, and 3 form a group under addition, but where we take the addition to be “remainder 4”, so that e.g. $3 + 2 = 1$. In abstract algebra, this group is denoted by C_4 .

The **dihedral group of order 8** (sometimes, very confusingly, also referred to as being of order 4) arises as the symmetry group of a square: we can rotate a square by 90° , or reflect it along either of the axis joining any two diagonally opposite corners, or reflect it along either of the axis joining the midpoints of any two opposite sides — doing any of these actions leaves the square looking exactly how it started. But some of these actions describe the same thing! For example, reflecting through the vertical axis and then the horizontal axis is the same as rotating by 180° (try visualising this by flipping and rotating your hand!), which is a specific example of the more general fact (which we briefly touched upon in Section 2.12) that the composition of two reflections is the same as a rotation through twice the angle between the two axes. In abstract algebra, this group is denoted by D_8 (though in geometry it is often written as D_4 instead).

The relevance to the Pauli group is this: the central product of C_4 and D_8 is exactly \mathcal{P}_1 .

If it is clear that we are working with *tensor products* of Pauli matrices, then we often (as per usual) omit the tensor product symbol, writing e.g. $XY\mathbf{1}Z$ instead of $X \otimes Y \otimes \mathbf{1} \otimes Z$ when talking about \mathcal{P}_4 . Note however that we only do this when it is

obvious what we mean: this is very different from the product $XY\mathbf{1}Z = i\mathbf{1}$ inside $\mathcal{P}_1!$

Let's now talk a little bit about the algebraic structure of \mathcal{P}_n . Multiplying together elements is fairly simple: since they are tensor products, we multiply them component-wise, but just remembering to pay attention to the global phase. For example, we can multiply $ZXX\mathbf{1}$ and $XXYY$ in \mathcal{P}_4 as follows:

$$\begin{aligned}(ZXX\mathbf{1}) \cdot (XXYY) &= (ZX)(XX)(XY)(\mathbf{1}Y) \\ &= (iY)(\mathbf{1})(iZ)(Y) \\ &= -Y\mathbf{1}ZY.\end{aligned}$$

Next, any pair of elements in \mathcal{P}_n either commute or anticommute: given $P = P_1 \dots P_n$ and $Q = Q_1 \dots Q_n$, we notice that they commute exactly whenever the number of *anticommuting* components (indices j such that $P_j Q_j = -Q_j P_j$) is even, since then the minus signs cancel out. In other words, $PQ = (-1)^J QP$, where $J = |\{j \text{ such that } P_j Q_j = -Q_j P_j\}|$. For example, if we consider two elements of \mathcal{P}_9 and write \checkmark to mean that two components commute, and $!$ to mean that they don't, we can then just count to see if there are an odd or even number of $!$ overall, like so:

$$\begin{array}{cccccccccc} Z & X & Y & X & Y & Z & X & X & Y \\ Z & X & \mathbf{1} & Z & Z & X & \mathbf{1} & Y & Z \\ \hline \checkmark & \checkmark & \checkmark & ! & ! & ! & \checkmark & ! & ! \end{array}$$

and since there are 5 anticommuting components, we see that $ZXYXYZXXY$ and $ZX\mathbf{1}ZZX\mathbf{1}YZ$ anticommute.

Finally, the square of any element in \mathcal{P}_n is ± 1 . Indeed, all the elements in the Pauli group are unitary, and each one is either *Hermitian* (overall phase ± 1 and squares to 1) or *anti-Hermitian* (overall phase $\pm i$ and squares to -1). As per usual, we are only really interested in working with the Hermitian elements, and we refer to *these* as the Pauli operators.

An **n -qubit Pauli operator** is a Hermitian element of the n -qubit Pauli group \mathcal{P}_n .

Not only do elements of \mathcal{P}_n have eigenvalues equal to ± 1 , these eigenvalues must be of the same degeneracy, and the eigenspaces corresponding to each eigenvalue are of the same dimension, as we can see by taking the trace:

$$\mathrm{tr}(P_1 \otimes P_2 \otimes \dots \otimes P_n) = (\mathrm{tr} P_1)(\mathrm{tr} P_2) \dots (\mathrm{tr} P_n)$$

which is zero, except in the trivial case where $P_1 = P_2 = \dots = P_n = \mathbf{1}$. Last but not least, the n -qubit Pauli group spans the space of $(2^n \times 2^n)$ complex matrices.

7.2 Pauli stabilisers

The stabiliser (or stabilizer, if you like) formalism is an elegant technique that is often used to describe vectors and subspaces. Suppose you want to specify a particular vector in a Hilbert space. The most conventional way to do this would be to pick a basis and then list the coordinate components of the vector. But we could instead list a set of operators that leave this vector invariant. More generally, we can define a vector subspace (rather than just a single vector, which corresponds to a 1-dimensional subspace: its span) by giving a list of operators that fix this subspace. Such operators are called **stabilisers**.

We say that an operator S **stabilises** a (non-zero) state $|\psi\rangle$ if $S|\psi\rangle = |\psi\rangle$, and we then call $|\psi\rangle$ a **stabiliser state**. We say that S stabilises a subspace V if S stabilises every state in V , and we call the largest subspace V_S that is stabilised by S the **stabiliser subspace**.

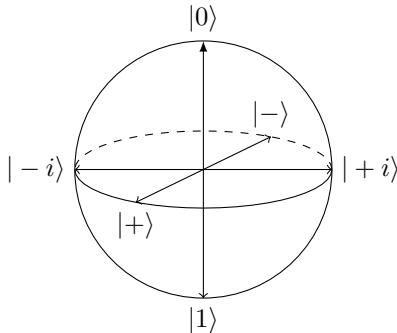
In other words, an operator S stabilises a state $|\psi\rangle$ (or the state is fixed by the operator) if $|\psi\rangle$ is an eigenstate of S with eigenvalue 1. It is very important to note that here we *have* to pay attention to the global phase factor: if $S|\psi\rangle = -|\psi\rangle$ then we *do not* say that S stabilises $|\psi\rangle$, even though $|\psi\rangle$ and $-|\psi\rangle$ describe the same quantum state.

For example, we can look at states stabilised by the Pauli operators with factors ± 1 :

$$\begin{array}{ll} Z \text{ stabilises } |0\rangle & -Z \text{ stabilises } |1\rangle \\ Y \text{ stabilises } |i\rangle & -Y \text{ stabilises } |-i\rangle \\ X \text{ stabilises } |+\rangle & -X \text{ stabilises } |-\rangle \end{array}$$

where $|\pm i\rangle = \frac{1}{\sqrt{2}}(|0\rangle \pm i|1\rangle)$ and $|\pm\rangle = \frac{1}{\sqrt{2}}(|0\rangle \pm |1\rangle)$.

On the Bloch sphere, these single-qubit stabiliser states lie at the intersection of the three axes with the surface of the sphere.



We can also say something about the remaining two elements of the single-qubit Pauli group: $\mathbf{1}$ stabilises everything, and $-\mathbf{1}$ stabilises nothing (except for the zero state, which we explicitly ignore). More generally, if S stabilises something then $-S$ cannot stabilise the same thing.

The set of all stabilisers of a given state or given subspace form a group: if $S|\psi\rangle = |\psi\rangle$, then multiplying both sides by S^{-1} shows that the inverse of a stabiliser is again a stabiliser; the composition of two stabilisers is again a stabiliser, since $(ST)|\psi\rangle = S(T|\psi\rangle) = S|\psi\rangle = |\psi\rangle$; and as we have just said, the identity is always a stabiliser. This group is called the **stabiliser group** S of the given state or subspace.

Using this language, we can rephrase the previous example by saying that the stabiliser group of the state $|0\rangle$ is $\{\mathbf{1}, Z\} = \langle Z \rangle$, the stabiliser group of the state $|1\rangle$ is $\{\mathbf{1}, -Z\} = \langle -Z \rangle$, the stabiliser group of the state $|+\rangle$ is $\{\mathbf{1}, X\} = \langle X \rangle$, and so on. If we take the tensor product of a two states, with stabiliser groups \mathcal{A} and \mathcal{B} (respectively), then the resulting tensor product state has stabiliser group given by the cartesian product $\mathcal{A} \times \mathcal{B}$. For example, the state $|1\rangle|+\rangle$ is stabilised by the group

$$\begin{aligned} \{\mathbf{1}, Z\} \times \{\mathbf{1}, X\} &= \{\mathbf{1}\mathbf{1}, \mathbf{1}X, Z\mathbf{1}, ZX\} \\ &= \langle Z\mathbf{1}, \mathbf{1}X \rangle. \end{aligned}$$

As for the state $|0\rangle^{\otimes n}$, this is stabilised by the group generated by the n elements $Z\mathbf{1}\dots\mathbf{1}, \mathbf{1}Z\mathbf{1}\dots\mathbf{1}, \dots, \mathbf{1}\mathbf{1}\dots, Z$, so we often simply stack the generators and write

such generating sets as $(n \times n)$ matrices, labelling the left-hand side with the relevant signs:

$$|0000\rangle \longleftrightarrow \begin{array}{c} + \\ + \\ + \\ + \end{array} \left| \begin{array}{cccc} Z & 1 & 1 & 1 \\ 1 & Z & 1 & 1 \\ 1 & 1 & Z & 1 \\ 1 & 1 & 1 & Z \end{array} \right|$$

and we can see that the signs determine the bit value in the computational basis state, if we look at the generators of the stabiliser groups for some other states:

$$|0001\rangle \longleftrightarrow \begin{array}{c} + \\ + \\ + \\ - \end{array} \left| \begin{array}{cccc} Z & 1 & 1 & 1 \\ 1 & Z & 1 & 1 \\ 1 & 1 & Z & 1 \\ 1 & 1 & 1 & Z \end{array} \right| \quad |0101\rangle \longleftrightarrow \begin{array}{c} + \\ - \\ + \\ - \end{array} \left| \begin{array}{cccc} Z & 1 & 1 & 1 \\ 1 & Z & 1 & 1 \\ 1 & 1 & Z & 1 \\ 1 & 1 & 1 & Z \end{array} \right|$$

For our purposes, we are only really interested in stabilisers that are also elements of the n -qubit Pauli group \mathcal{P}_n , and we shall soon see that these form an *abelian* group. It turns out that such stabilisers can describe highly entangled states. In particular, the four Bell states (which we first talked about in Section 5.7) can be defined rather succinctly by their stabiliser groups:

Bell state	Stabiliser group
$\Phi^+ = 00\rangle + 11\rangle$	$\langle XX, ZZ \rangle$
$\Psi^+ = 01\rangle + 10\rangle$	$\langle XX, -ZZ \rangle$
$\Phi^- = 00\rangle - 11\rangle$	$\langle -XX, ZZ \rangle$
$\Psi^- = 01\rangle - 10\rangle$	$\langle -XX, -ZZ \rangle$

Not only this, but some vector spaces are also rather easily defined: the subspace of the three-qubit state space spanned by $|000\rangle$ and $|111\rangle$ is stabilised by

$$\{111, ZZ1, Z1Z, 1ZZ\} = \langle ZZ1, 1ZZ \rangle.$$

Right now, it might seem more complicated to use stabilisers to define vectors or subspaces, but when we start looking at states with a larger and larger number of components we will see how this approach ends up being very tidy indeed! It is not true that the stabiliser description of states and subspaces will *always* be the most concise, but it is true in a lot of cases that are of interest to us.

Returning to our claim that stabiliser groups that are subgroups of \mathcal{P}_n are abelian, let us start with a definition, and then justify it afterwards.

An **n -qubit Pauli stabiliser group** is any subgroup of \mathcal{P}_n that is abelian and does not contain -1 . Its elements are called **Pauli stabilisers**.

Recall that, in order for the subspace V_S stabilised by some group S to be non-trivial, we need $-1 \notin S$. Given that all Pauli operators square to the identity, and all pairs of Pauli operators either commute or anticommute, this implies that if we want some Pauli operators to stabilise anything then they must *commute*. Indeed, if S_1 and S_2 are two Pauli operators that anticommute, and $|\psi\rangle$ is any vector stabilised by both of them, then

$$\begin{aligned} |\psi\rangle &= S_1 S_2 |\psi\rangle \\ &= -S_2 S_1 |\psi\rangle \\ &= -|\psi\rangle \end{aligned}$$

which means that $|\psi\rangle = 0$. But saying that we are looking at a stabiliser group consisting of Pauli stabilisers that all commute with one another (as opposed to anti-commuting) is exactly saying that we have an abelian subgroup of \mathcal{P}_n ; if we want it to be non-trivial, then we need it to not contain -1 . Conversely, if we pick any abelian subgroup of \mathcal{P}_n that does not contain -1 , this stabilises *some* subspace V_S .

The size of any Pauli stabiliser S is $|S| = 2^r$, where r is some positive integer, since we can always find some choice of generators G_1, \dots, G_r , and then any operator $S \in S$ can be written as

$$S = G_1^{\epsilon_1} G_2^{\epsilon_2} \dots G_r^{\epsilon_r}$$

where $\epsilon_i \in \{0, 1\}$. But given any stabiliser group, we can always express its elements using many different sets of generators; a specific choice of r independent generators of a Pauli stabiliser S of size 2^r is called a **presentation**. In order to choose a presentation from the set of elements of S , we have to start by picking any non-identity element, of which there are $2^r - 1$. Inductively then, we pick the next generator by picking any element which is not in the subgroup generated by the previously selected generators, which means that there are

$$(2^r - 1)(2^r - 2)(2^r - 2^2) \dots (2^r - 2^{r-1})$$

possible generating sets of S . But these are *ordered* sets (i.e. we are keeping track of the order in which we pick the elements, so G_1, G_2, \dots is a “different” choice than G_2, G_1, \dots), so if we want to know the number of presentations then we can simply divide the expression above by $r!$.

For example, the Bell state $\Phi^+ = |00\rangle + |11\rangle$ is stabilised by the group $\{11, XX, -YY, ZZ\}$. This stabiliser group has $(2^2 - 1)(2^2 - 2)/2! = 3$ presentations, namely $\langle XX, ZZ \rangle$, $\langle -YY, XX \rangle$, and $\langle ZZ, -YY \rangle$.

So now we know the size of a Pauli stabiliser, but what can we say about the dimension of the subspace that it stabilises? If $|S| = 2^r$ then the corresponding stabiliser subspace V_S has dimension 2^{n-r} (where n is the number of qubits, i.e. such that $S \subseteq \mathcal{P}_n$). To see this, we can look at the projector P_S onto V_S , since once we have a projector onto any subspace we know that the dimension of that subspace is exactly the trace of the projector (we can prove this by thinking about the matrix of the projector in the diagonal form). In our case (using the result of Exercise 7.8.5) we calculate that

$$\begin{aligned} \text{tr } P_S &= \text{tr } \frac{1}{2^r} (S_1 + S_2 + \dots + S_{2^r}) \\ &= \frac{1}{2^r} (\text{tr } \mathbf{1}) \\ &= 2^{n-r} \end{aligned}$$

since any non-identity element of the stabiliser group has trace equal to zero, and $\text{tr } \mathbf{1}^{\otimes n} = 2^n$, whence $\dim V_S = 2^{n-r}$. If $r = n$ then the stabilised subspace is 1-dimensional, and so we have stabiliser states.

There is a more geometric way of understanding why powers of 2 keep on turning up in these calculations. Given independent Pauli generators, it is convenient to think about the state or subspace that they stabilise as being the result of repeatedly bisecting the Hilbert space. Let G_1, \dots, G_r be a presentation of a Pauli stabiliser S . For each operator G_i , half its eigenvalues are $+1$ and another half are -1 , so each G_i bisects the 2^n -dimensional Hilbert space of n qubits into two eigenspaces of equal size. So G_1 gives two 2^{n-1} -dimensional subspaces: one for the $+1$ eigenvalue and one for the -1 eigenvalue. Forgetting about the -1 part and just focusing on the $+1$ part, G_2 then splits this 2^{n-1} -dimensional subspace into two 2^{n-2} -dimensional subspaces, since it is independent from G_1 (as we justify in Exercise 7.8.5). Repeating this procedure, forgetting about the -1 subspace each time, leads us to consider the simultaneous $+1$ -eigenspace of G_1, \dots, G_r , where each time we pass from $\{G_1, G_2, \dots, G_i\}$ to

An interesting small exercise here is to explain why the product of any *independent* Pauli stabilisers cannot be equal to the identity.

$\{G_1, G_2, \dots, G_i, G_{i+1}\}$ we bisect the subspace into two equal parts once more, eventually ending with the 2^{n-2} -dimensional subspace V_S , as above. We can show this pictorially, as in Figure 7.1.

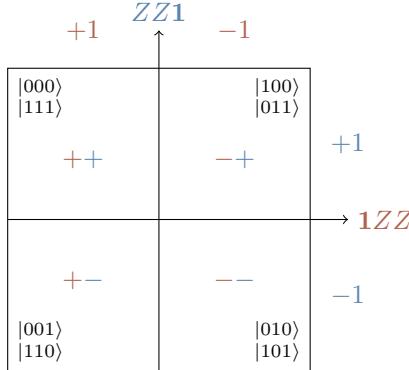


Figure 7.1: The stabiliser group $S = \langle ZZ1, 1ZZ \rangle$ bisects the Hilbert space of three qubits into four equal parts, and gives the stabilised subspace V_S which is spanned by $|000\rangle$ and $|111\rangle$. Think of the labels $ZZ1$ and $1ZZ$ as the x - and y -axes, and the sign labels on each square as (x, y) -coordinates. So the two squares on the left together make the $+1$ -eigenspace of $1ZZ$, and the two squares on the top make the $+1$ -eigenspace of $ZZ1$.

This diagram will make a reappearance in Sections 13 and 14.

7.3 Single stabiliser states

Given n independent generators of a stabiliser group S on a Hilbert space of n -qubits, we end up specifying a 1-dimensional subspace, meaning it is spanned by a single basis vector, namely the stabiliser state. We have already talked about the single-qubit stabiliser states determined by all possible stabilisers in \mathcal{P}_1 , namely $|0\rangle$ and $|1\rangle$ for $\langle \pm Z \rangle$, $|\pm\rangle$ for $\langle \pm X \rangle$, and $|\pm i\rangle$ for $\langle \pm Y \rangle$. We have also mentioned some of the two-qubit stabilisers states, some of which are highly entangled, such as the Bell states, and some of which are separable, such as the computational basis states (whose stabilisers groups we described by block matrices with Z on the diagonal, 1 everywhere else, and signs labelling each row depending on the binary description of the state).

Here's another two-qubit example: that of the maximally entangled state $|00\rangle + |11\rangle$. This is stabilised by $\langle XX, ZZ \rangle$, but let's explain how we can see this. If we look first at the operator XX , we see that it splits the 4-dimensional Hilbert space into two 2-dimensional subspaces, corresponding to eigenvalues ± 1 ; by definition, it stabilises the one corresponding to eigenvalue $+1$, which is spanned by $|00\rangle + |11\rangle$ and $|01\rangle + |10\rangle$. Now the operator ZZ also splits the 4-dimensional Hilbert space into two 2-dimensional subspaces, again corresponding to eigenvalues ± 1 ; it stabilises the one corresponding to eigenvalue $+1$, which is spanned by $|00\rangle + |11\rangle$ and $|00\rangle - |11\rangle$. Note that $|01\rangle + |10\rangle$ is in the -1 -eigenspace of ZZ , even though it is in the $+1$ -eigenspace of XX (and vice versa for $|00\rangle - |11\rangle$). So the simultaneous $+1$ -eigenspace of XX and ZZ is exactly the state $|00\rangle + |11\rangle$.

$$\begin{array}{lcl} |00\rangle + |11\rangle \longleftrightarrow + \left| \begin{array}{cc} X & X \\ Z & Z \end{array} \right| & |00\rangle - |11\rangle \longleftrightarrow - \left| \begin{array}{cc} X & X \\ Z & Z \end{array} \right| \\ |01\rangle + |10\rangle \longleftrightarrow + \left| \begin{array}{cc} X & X \\ Z & Z \end{array} \right| & |01\rangle - |10\rangle \longleftrightarrow - \left| \begin{array}{cc} X & X \\ Z & Z \end{array} \right| \end{array}$$

As we have already mentioned when discussing presentations of a stabiliser group, there can be multiple different generating sets, which corresponds to the fact that

there are multiple different ways of bisecting the Hilbert space. For example, the stabiliser state $|00\rangle + |11\rangle$ is completely specified by $\langle XX, ZZ \rangle$, as shown above, but also by $\langle XX, -YY \rangle$ or $\langle -YY, ZZ \rangle$. But, as we should expect, these three generating sets all generate the same group, namely $\mathcal{S} = \{11, XX, -YY, ZZ\}$.

How many n -qubit stabiliser states do we have? The answer is

$$2^n \prod_{k=0}^{n-1} (2^{n-k} + 1)$$

as we can show with a counting argument: we will count the number of generating sets with n generators (since this is exactly the right number of generators to specify a 1-dimensional stabiliser subspace) and then divide by the number of presentations for any given stabiliser. There are 4^{n-1} choices for the first generator G_1 (ignoring overall sign), since it can be any n -fold tensor product of the four Pauli matrices, excluding the identity 1111 . For the second generator G_2 , we have $(4^n/2) - 2$ possibilities, since it must commute with the first generator (and we know that exactly half of the operators commute with any given operator, as shown in Exercise 7.8.3, whence $4^n/2$) and it cannot be 1111 or G_1 (whence -2). Similarly, G_3 must commute with both G_1 and G_2 , but it cannot be in the group generated by them, so there are $(4^n/4) - 4$ possible choices, and so on. This means that we have

$$2^n(4^n - 1) \left(\frac{4^n}{2} - 2\right) \left(\frac{4^n}{4} - 4\right) \dots \left(\frac{4^n}{2^{n-1}} - 2^{n-1}\right)$$

possible generating sets in total. Now we need to divide by the number of presentations, but we have already calculated this in Section 7.2: it's exactly

$$(2^n - 1)(2^n - 2)(2^n - 2^2) \dots (2^n - 2^{n-1}).$$

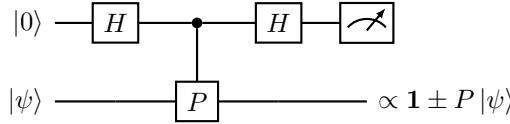
It is a fun algebra exercise to show that this division indeed gives the number we claimed.

As we will see, stabiliser states are ubiquitous in quantum information theory due to their versatility and relative simplicity. They play a crucial role in areas such as quantum error correction, measurement-based quantum computation, and entanglement classification.

7.4 Measuring Pauli stabilisers

How do we bisect Hilbert spaces in practice? By measuring stabilisers.

Let's start by measuring any single-qubit observable that squares to the identity. The corresponding operator P with eigenvalues ± 1 is both Hermitian and unitary, and can thus represent both an observable and a quantum gate. If we prepare a qubit in some state $|\psi\rangle$ and then wish to perform a measurement that will give us a result of ± 1 and leave the qubit in a post-measurement state, namely the corresponding eigenvector, then we can use the following circuit (where \propto denotes that two states are multiples of one another).



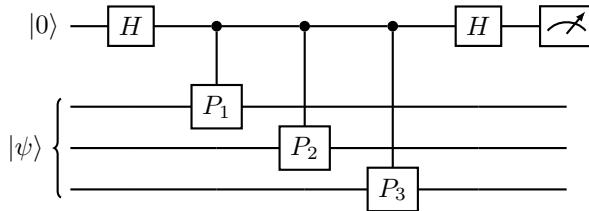
This construction requires an auxiliary qubit (in the top register), two Hadamard gates, and the tacit assumption that we can construct a controlled- P operator. Step-

This is a common technique in combinatorial arguments: first overcount, and then fix your answer by accounting for this.

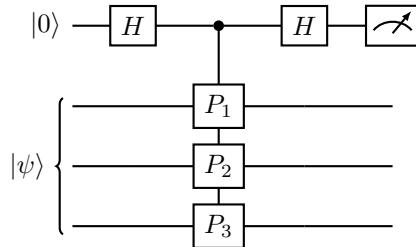
ping through the execution of this circuit, we get

$$\begin{aligned} |0\rangle|\psi\rangle &\xrightarrow{H \otimes \mathbf{1}} \frac{1}{\sqrt{2}}(|0\rangle + |1\rangle)|\psi\rangle \\ &\xrightarrow{\text{c-P}} \frac{1}{\sqrt{2}}|0\rangle|\psi\rangle + \frac{1}{\sqrt{2}}|1\rangle P|\psi\rangle \\ &\xrightarrow{H \otimes \mathbf{1}} |0\rangle \frac{1}{2}(\mathbf{1} + P)|\psi\rangle + |1\rangle \frac{1}{2}(\mathbf{1} - P)|\psi\rangle. \end{aligned}$$

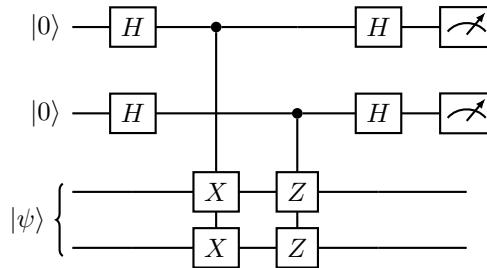
The final state of the two qubits indicates that, when the auxiliary (top) qubit is found in state $|0\rangle$ then we projected the state $|\psi\rangle$ onto the $+1$ -eigenspace of P (via the projector $\frac{1}{2}(\mathbf{1} + P)$), and when it is found in state $|1\rangle$ then we projected $|\psi\rangle$ onto the -1 -eigenspace (via the projector $\frac{1}{2}(\mathbf{1} - P)$). In particular, the X , Y , and Z observables can be measured using controlled- X , controlled- Y , and controlled- Z gates (respectively). This pattern can easily be extended to an n -qubit Pauli operator. For example, for $n = 3$, a generic circuit that implements a projective measurement onto the ± 1 -eigenspaces of $S = P_1 \otimes P_2 \otimes P_3$ has the form



and is usually drawn more compactly as



In this way, we can measure stabilisers and project onto the subspaces that they stabilise. For example, take the stabiliser group $S = \langle XX, ZZ \rangle$, and consider the circuit below:



The registered bit values from the first and second (counting from the top) auxiliary qubits tell us how we bisect the Hilbert space with XX and ZZ (respectively), recalling that a bit value of 0 corresponds to the $+1$ Pauli eigenvalue, and a bit value of 1 to the -1 eigenvalue. The first measurement can apply one of two projectors to $|\psi\rangle$:

- a. $\frac{1}{2}(1 + XX)$, in which case the first auxiliary qubit will show 0, corresponding to the eigenvalue +1, and the subspace spanned by $|00\rangle + |11\rangle$ and $|01\rangle + |10\rangle$
- b. $\frac{1}{2}(1 - XX)$, in which case the first auxiliary qubit will show 1, corresponding to the eigenvalue -1, and the subspace spanned by $|00\rangle - |11\rangle$ and $|01\rangle - |10\rangle$.

The second measurement can further project the resulting post-measurement state of the two qubits in one of two ways:

- a. $\frac{1}{2}(1 + ZZ)$, in which case the second auxiliary qubit will show 0, corresponding to the eigenvalue +1, and the subspace spanned by $|00\rangle + |11\rangle$ and $|00\rangle - |11\rangle$
- b. $\frac{1}{2}(1 - ZZ)$, in which case the second auxiliary qubit will show 1, corresponding to the eigenvalue -1, and the subspace spanned by $|01\rangle + |10\rangle$ and $|01\rangle - |10\rangle$.

So if both auxiliary qubits show bit value outcome 0 (corresponding to the **Pauli outcome** $(+1, +1)$ of eigenvalues), then we have successfully projected onto the state stabilised by XX and ZZ , which is exactly $|00\rangle + |11\rangle$. More generally, in **Pauli notation**, the outcome $(\pm 1, \pm 1)$ corresponds to the projection onto the stabiliser state stabilised by $\langle \pm XX, \pm ZZ \rangle$.

Needless to say, we do not have any control over the actual outcomes of the measurement, but we do now know *which* post-measurement state we have generated. This means that we can use the circuit to prepare a desired state by applying an appropriate unitary operation to the final state. For example, if we want to generate the state $|00\rangle + |11\rangle$ but actually end up with the state $|00\rangle - |11\rangle$, then we can simply apply the Z operation to any of the two qubits to get the desired result. This generic method is not the only way of constructing projective measurements of Pauli observables, however — see Exercise 7.8.7

7.5 Normal subgroups

Before continuing our exploration of Pauli stabilisers, we need a bit more abstract mathematics.

Let H be a subgroup of G , written $H \leq G$. We say that H is a **normal** subgroup of G , and write $H \triangleleft G$, if H is invariant under conjugation by all elements of G , i.e. $ghg^{-1} \in H$ for all $g \in G$ and all $h \in H$.

Note that we only require that ghg^{-1} be some arbitrary element in H , not that $ghg^{-1} = h$.

If $H \leq G$ is an arbitrary (not necessarily normal) subgroup, then we can use it to “slice up” G into subsets of equal size called **cosets**, one of which is H itself. We define a (left) coset to be a set of the form

$$gH = \{gh \mid h \in H\}$$

for any fixed $g \in G$. Any two cosets (i.e. any two choices of $g \in G$) are either entirely equal or completely disjoint. The relevance of normality here is that if H is normal, then there is no need to distinguish between left (gH) and right (Hg) cosets, and in this case we can construct the **quotient group** G/H consisting of cosets with the operation defined by $gH \cdot g'H = (gg')H$. Here is one way to visualise a partition into cosets:

$$\begin{array}{l} c_nH = \begin{vmatrix} c_n & c_nh_1 & \dots & c_nh_k \\ \vdots & \vdots & \ddots & \vdots \end{vmatrix} \\ c_2H = \begin{vmatrix} c_2 & c_2h_1 & \dots & c_2h_k \\ c_1H = \begin{vmatrix} c_1 & c_1h_1 & \dots & c_1h_k \\ H = \begin{vmatrix} \mathbf{1} & h_1 & \dots & h_k \end{vmatrix} \end{vmatrix} \end{array}$$

The bottom row represents the subgroup H , and each row above represents a coset, i.e. a set of elements generated by picking an element c_k of G that does not belong to H nor to any of the previously generated cosets, and then multiplying this element by all elements in H , one at a time. This picture above shows that, for any finite group G and any subgroup $H \leq G$,

$$|G| = |G : H| \cdot |H|$$

where $|G : H|$ is the number of cosets of G given by H . This fact is known as **Lagrange's theorem** (although [Joseph-Louis Lagrange](#) was a rather prolific mathematician, working in many areas, so this is only one of the theorems to bear his name).

It seems like it was [Évariste Galois](#) who recognised that *normal* subgroups were worthy of special attention. Given an arbitrary subgroup $H \leq G$, we can construct a larger subgroup $K \leq G$ in which H is normal, i.e. such that $H \triangleleft K \leq G$. The largest such subgroup K is called the **normaliser of H in G** , denoted by $N_G(H)$, and we can construct it explicitly:

$$N_G(H) = \{g \in G \mid ghg^{-1} \in H \text{ for all } h \in H\}.$$

In words, the normaliser consists of the set of elements of G that conjugate all elements of H to elements of H . This suggests a very subtle question: is every subgroup of a normal subgroup normal? The answer is most definitely *no*: if $H \triangleleft K$ and $K \triangleleft G$ then it is *not* necessarily the case that $H \triangleleft G$, merely that $H \leq G$.

As we shall soon see, Pauli stabilisers are not normal subgroups of \mathcal{P}_n , and we will instead want to study their normalisers.

7.6 Pauli normalisers

There are two subgroups that pop up once we choose a stabiliser S . The subgroup of \mathcal{P}_n consisting of all elements that commute with every element of S is called the **centraliser of S** , denoted by

$$Z(S) = \{g \in \mathcal{P}_n \mid gsg^{-1} = s \text{ for all } s \in S\}$$

and the other is the one that we have already seen: the normaliser

$$N(S) = \{g \in \mathcal{P}_n \mid gsg^{-1} \in S \text{ for all } s \in S\}.$$

These two are, in general, distinct but related: for the normaliser we ask that $gsg^{-1} = s'$ for some arbitrary $s' \in S$, and for the centraliser we *additionally* ask that $s' = s$. However, in the case of Pauli groups, these two subgroups coincide, because $gsg^{-1} = s'$ if and only if $sg = gs'$, but since any two elements of the Pauli group either commute or anticommute, this implies that $s' = s$ or $s' = -s$; but we have already seen (and proven, in Exercise 7.8.4) that if s is in a stabiliser then $-s$ cannot be, and so it must be the case that $s' = s$.

In summary, given a stabiliser, we get a corresponding normaliser, and given the normaliser (which, in our case, is the same thing as the centraliser), we get two interesting quotient groups to study. By the definition of the normaliser, S is normal in $N(S)$. What is less obvious is that $N(S)$ itself is normal in \mathcal{P}_n . To prove this, let $g \in \mathcal{P}_n$ and consider gng^{-1} , for some $n \in N(S)$. We need to show that $gng^{-1} \in N(S)$. For any $s \in S$,

$$(gng^{-1})s = s(gng^{-1})$$

because either s commutes with both g and g^{-1} , or it anticommutes with both g and g^{-1} , in which case the two minus signs cancel out. Either way, s commutes with gng^{-1} , and so, by definition, $gng^{-1} \in N(S)$. So S is normal in $N(S)$ by definition, and $N(S)$ is normal in \mathcal{P}_n by the above; but recall that this does *not* imply that S is

When the ambient group G is evident, we often simply denote the normaliser by $N(H)$. But the choice of the group G still matters!

The letter Z stands for the German *Zentrum*, which means *centre*.

normal in \mathcal{P}_n . Indeed, pick any element $g \in \mathcal{P}_n$ that anticommutes with some $s \in \mathcal{S}$, so that $gsg^{-1} = -s$; but we already know that if $s \in \mathcal{S}$ then $-s \notin \mathcal{S}$.

With these normal subgroups $\mathcal{S} \triangleleft N(\mathcal{S}) \triangleleft \mathcal{P}_n$ we can form two quotient groups, arranging things into cosets: $N(\mathcal{S})/\mathcal{S}$ and $\mathcal{P}_n/N(\mathcal{S})$. This is visualised in Figure 7.2. Note that we can also form cosets of \mathcal{S} in \mathcal{P}_n , but since this is not a normal subgroup the left cosets will be different from the right cosets, and we cannot construct a quotient group $\mathcal{P}_n/\mathcal{S}$.

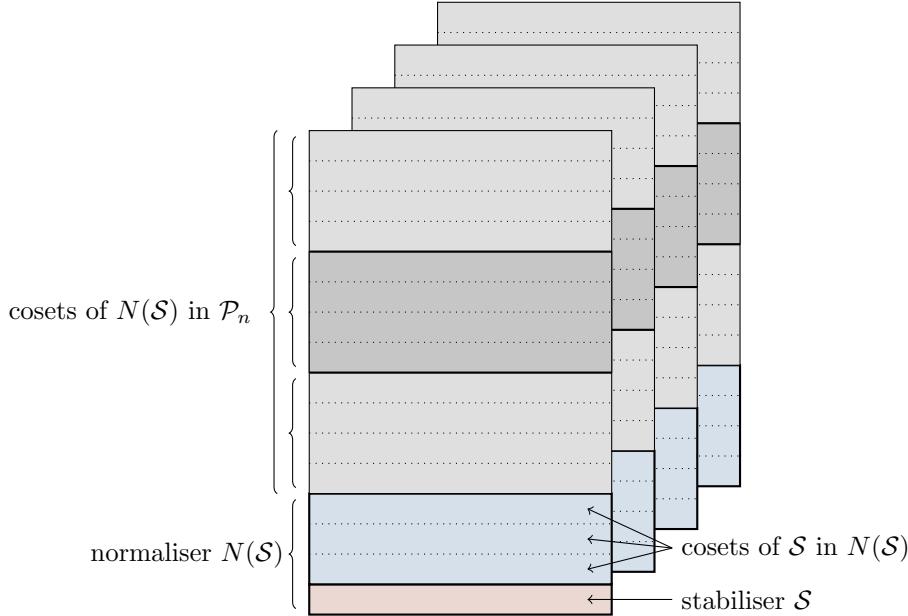


Figure 7.2: Any Pauli stabiliser $\mathcal{S} \leqslant \mathcal{P}_n$ (shown in red) slices its normaliser $N(\mathcal{S})$ (shown in blue) into cosets, and the normaliser in turn slices \mathcal{P}_n into cosets. There are four “sheets” in the diagram to remind us that there are four possible global phases, so we obtain the three sheets behind the first one by multiplying by -1 , i , and $-i$. We give a more concrete worked example in Exercise 7.8.2.

This resulting structure — any Pauli stabiliser slicing its normaliser into cosets, and this normaliser in turn slicing the Pauli group into cosets — will be very useful when we discuss quantum error correction and fault tolerance in Sections 13 and 14, and we can explain a bit how this will work now. The stabiliser will partition the Hilbert space of n qubits into subspaces, and the one that is fixed by the stabiliser will be chosen as a **codespace**. All operators in the normaliser will then become **logical operators** on the codespace, and the cosets of the normaliser in \mathcal{P}_n will group together operators that describe errors of a similar type (those with the same **error syndrome**). It will be a useful fact to know that $\mathcal{P}_n/N(\mathcal{S})$ is abelian — we show this in Exercise 7.8.6.

Finally, let's count some elements. We have already seen that $|\mathcal{P}_n| = 4 \cdot 4^n = 4^{n+1}$, and that if S has r generators then $|S| = 2^r$. But what about $N(\mathcal{S})$? By definition, the normaliser consists of all the operators that commute with all the generators of the stabiliser. There are $4 \cdot (4^n/2)$ that commute with the first generator, half of which also commute with the second generator, a further half of which also commute with the third generator, and so on. So we have that $|N(\mathcal{S})| = 4 \cdot (4^n/2^r)$. Finally we have the two quotient groups: $N(\mathcal{S})/\mathcal{S}$ has $4 \cdot (4^n/4^r) = 4^{n-r+1}$ elements and is isomorphic to \mathcal{P}_{n-r} ; and $\mathcal{P}_n/N(\mathcal{S})$ has $(4 \cdot 4^n)/(4 \cdot (4^n/2^r)) = 2^r$ elements.

7.7 Clifford walks on stabiliser states

There are essentially two ways to define stabiliser states of n qubits. We have already seen how we can describe them as simultaneous $+1$ eigenstates of n generators of some stabiliser group $\mathcal{S} \leq \mathcal{P}_n$, but it turns out that we could also define them as the states that are reachable from the $|0\rangle^{\otimes n}$ state using only the c-NOT gate, the Hadamard H , and the phase gate $S = \begin{pmatrix} 1 & 0 \\ 0 & i \end{pmatrix}$. If you start playing around with these three gates, you'll soon notice that you tend to reach certain discrete states, and never anything in between them. For example, in the single qubit case (so with just the H and S gates), you'll be able to go between $|0\rangle$, $|1\rangle$, $|\pm\rangle$, and $|\pm i\rangle$, but never anything like, say, $\sqrt{\frac{1}{3}}|0\rangle + \sqrt{\frac{2}{3}}|1\rangle$. When you have two or more qubits, you might also notice that whenever you create an n -qubit superposition that assigns non-zero amplitudes to strings in some set $A \subset \{0, 1\}^n$, it's always an equal superposition over A (though possibly with ± 1 or $\pm i$ phases), and $|A|$ is always some power of 2. For example, you can generate states such as $\frac{1}{\sqrt{2}}(|000\rangle + |111\rangle)|010\rangle$ or $\frac{1}{2}(|000\rangle + i|100\rangle + |011\rangle - i|111\rangle)|010\rangle$, but never states such as $\frac{1}{\sqrt{3}}(|001\rangle + |010\rangle + |100\rangle)|010\rangle$.

Circuits composed of only c-NOT, H , and $S = P_{\pi/2}$ are special: they effect unitaries that map stabiliser states to stabiliser states.

The n -qubit **Clifford group** \mathcal{C}_n is the group generated by these three unitaries, and it happens to be exactly the normaliser of the n -qubit Pauli group inside the group of all $(2^n \times 2^n)$ unitary matrices:

$$\mathcal{C}_n = \{U \in U(2^n) \mid UPU^\dagger \in \mathcal{P}_n \text{ for all } P \in \mathcal{P}_n\} =: N_{U(2^n)}(\mathcal{P}_n).$$

It's a confusing (but immutable) matter of terminology that **Clifford gates** (i.e. gates made from only unitaries in the Clifford group) are sometimes called **stabiliser gates**, and **Clifford circuits** (i.e. circuits made from only Clifford gates) are sometimes called **stabiliser circuits**, but stabiliser states are *never* called "Clifford states".

So if we have an n -qubit stabiliser state, described by n Pauli generators, then any unitary in the Clifford group \mathcal{C}_n will map each of the n Pauli generators to another Pauli generator, and the set of these n new generators will define a new stabiliser state. Indeed, suppose we have some vector space V stabilised by the group \mathcal{S} , and we apply some unitary operation U . If $|\psi\rangle$ is an arbitrary element of V , then, for any element S of \mathcal{S} ,

$$\begin{aligned} U|\psi\rangle &= US|\psi\rangle \\ &= US(U^\dagger U)|\psi\rangle \\ &= (USU^\dagger)U|\psi\rangle \end{aligned}$$

and so the state $U|\psi\rangle$ is stabilised by USU^\dagger , from which we deduce that the vector space

$$UV := \{U|\psi\rangle \mid |\psi\rangle \in V\}$$

is stabilised by the group

$$USU^\dagger := \{USU^\dagger \mid S \in \mathcal{S}\}.$$

Furthermore, if G_1, \dots, G_r generate \mathcal{S} , then $UG_1U^\dagger, \dots, UG_rU^\dagger$ generate USU^\dagger , so to compute the change in the stabiliser we need only compute how it affects the generators of the stabiliser.

Since the Clifford group is generated by only three elements, we can easily work out how each of these gates acts by conjugation on the Pauli group. For instance, we

have previously seen that the Hadamard gate performs the following transformation:

$$X \mapsto HXH = Z$$

$$Z \mapsto HZH = X.$$

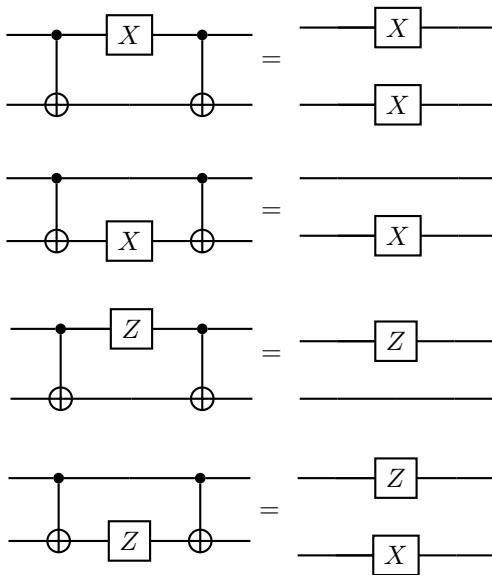
Given that $Y = iXZ$, there is no need to specify the action of H on Y , since we can calculate that

$$\begin{aligned} Y &\mapsto i(HXH)(HZH) \\ &= iZX \\ &= -Y. \end{aligned}$$

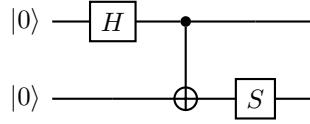
All the basic rules for updating stabilisers with Clifford gates can be conveniently tabulated:

Gate	Input/Output
H	$\left\{ \begin{array}{l} X \mapsto Z \\ Y \mapsto -Y \\ Z \mapsto X \end{array} \right\}$
S	$\left\{ \begin{array}{l} X \mapsto Y \\ Y \mapsto -X \\ Z \mapsto Z \end{array} \right\}$
c-NOT	$\left\{ \begin{array}{l} \mathbf{1}X \mapsto \mathbf{1}X \\ X\mathbf{1} \mapsto XX \\ \mathbf{1}Y \mapsto ZY \\ Y\mathbf{1} \mapsto YX \\ \mathbf{1}Z \mapsto ZZ \\ Z\mathbf{1} \mapsto Z\mathbf{1} \end{array} \right\}$

and these rules can be expressed as circuit identities, such as



Let's work through an example to see how these rules work in practice. Here's a simple stabiliser circuit:



As we step through this circuit we embark on our **Clifford walk** between two-qubit stabiliser states:

$$\begin{aligned} |00\rangle &\xrightarrow{H \otimes 1} \frac{1}{\sqrt{2}}(|0\rangle + |1\rangle)|0\rangle \\ &\xrightarrow{\text{c-NOT}} \frac{1}{\sqrt{2}}(|00\rangle + |11\rangle) \\ &\xrightarrow{1 \otimes S} \frac{1}{\sqrt{2}}(|00\rangle + i|11\rangle). \end{aligned}$$

This walk could also be described in terms of stabiliser generators:

$$\begin{vmatrix} Z & 1 \\ 1 & Z \end{vmatrix} \xrightarrow{H \otimes 1} \begin{vmatrix} X & 1 \\ 1 & Z \end{vmatrix} \xrightarrow{\text{c-NOT}} \begin{vmatrix} X & X \\ Z & Z \end{vmatrix} \xrightarrow{1 \otimes S} \begin{vmatrix} X & Y \\ Z & Z \end{vmatrix}.$$

Here the first column corresponds to the first qubit, and the second column to the second qubit. So the first Hadamard gate flips $Z1$ to $X1$, then the c-NOT (which acts on both qubits together, and so acts on entire rows) turns $X1$ into XX and $1Z$ into ZZ , then finally the S gate on the second qubit (thus the second column) turns XZ into YZ .

Despite the fact that the Clifford circuits can generate huge entangled n -qubit superpositions starting from the single state $|0\rangle^{\otimes n}$, such circuits are easy to simulate classically because we can efficiently update the list of stabilisers following the simple rules. Daniel Gottesman and Emanuel Knill showed that there is a polynomial-time classical algorithm to simulate any stabiliser circuit that acts on a stabiliser state. We can also efficiently compute the expectation values of any physical observables by examining the updated list of stabilisers. Note that computing a list of amplitudes would not be efficient, since there are exponentially many of them.

This is now known as the [Gottesman–Knill theorem](#).

Because they can be efficiently classically simulated, stabiliser circuits necessarily do *not* capture the full power of quantum computation. Fully universal quantum computation requires at least one non-Clifford gate, such as the $T = \begin{pmatrix} 1 & 0 \\ 0 & e^{-i\pi/4} \end{pmatrix}$ gate. Once we include this gate, we can create circuits that will take us from any initial state, such as $|0\rangle^{\otimes n}$, to arbitrarily close to any other state in the n -qubit Hilbert space. Despite this limitation of stabiliser computation, however, it has become a central part of quantum computing, mostly because of its role in quantum error correction and fault-tolerant computation. Almost all of the quantum error correcting codes are stabiliser codes, and are presented using the stabiliser formalism.

7.8 Remarks and exercises

7.8.1 Measuring parity

Suppose you have a two-qubit stabiliser ZZ . This is an observable that has two eigenvalues and two corresponding eigenspaces, namely the $+1$ -eigenspace spanned by $\{|00\rangle, |11\rangle\}$ and the -1 -eigenspace spanned by $\{|01\rangle, |10\rangle\}$. This tells us something about the parity of the two qubits: the $+1$ outcome means that the bit values are the same, and the -1 outcome means that they are different. However, it is critical that we do *not* measure the bit values $Z1$ and $1Z$ separately and then multiply the results, since this could cause the state to “collapse” to one of the basis states in revealing the exact bit values. We don’t want this! We simply want to know the mutual parity of the bit values, not what values they actually are.

Mathematically speaking, the parity measurement ZZ involves two orthogonal projectors

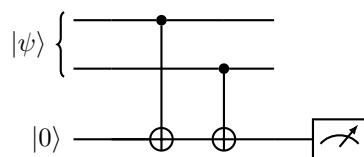
$$\frac{1}{2}(\mathbf{1}\mathbf{1} + ZZ) = |00\rangle\langle 00| + |11\rangle\langle 11|$$

$$\frac{1}{2}(\mathbf{1}\mathbf{1} - ZZ) = |01\rangle\langle 01| + |10\rangle\langle 10|$$

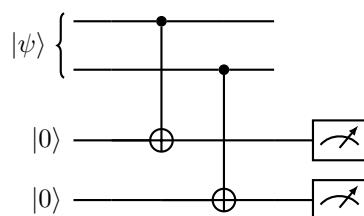
whereas the bit-value measurements $Z1$ and $1Z$ are characterised by the four projectors

$$\begin{aligned}\frac{1}{2}(\mathbf{1}\mathbf{1} + Z\mathbf{1}) &= |0\rangle\langle 0| \otimes \mathbf{1} \\ \frac{1}{2}(\mathbf{1}\mathbf{1} + \mathbf{1}Z) &= \mathbf{1} \otimes |0\rangle\langle 0| \\ \frac{1}{2}(\mathbf{1}\mathbf{1} - Z\mathbf{1}) &= |1\rangle\langle 1| \otimes \mathbf{1} \\ \frac{1}{2}(\mathbf{1}\mathbf{1} - \mathbf{1}Z) &= \mathbf{1} \otimes |1\rangle\langle 1|\end{aligned}$$

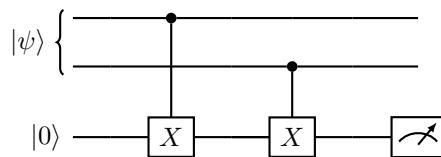
In terms of circuits, if we want to measure the parity of two qubits prepared in some state $|\psi\rangle$, then we can use the circuit



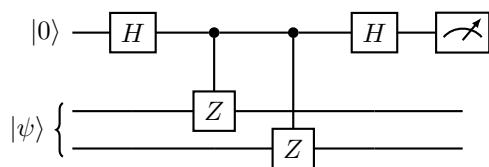
which is different from the circuit which effects the measurement of the individual bit values of the two qubits, namely



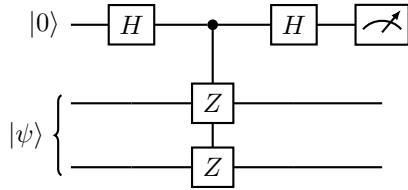
When dealing with stabilisers, we prefer to think of the controlled-NOT gate as a controlled- X instead. We then draw the parity-measurement circuit as



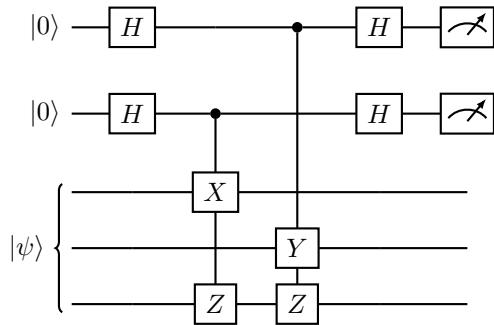
But this is a bit confusing, since we are measuring the Z observable using controlled- X gates. Thankfully, we can use some circuit/Pauli identities to rephrase things in terms of controlled- Z gates instead:



This is a quantum version of the two-bit parity measurement. When the auxiliary qubit (now in the top register) is found in state $|0\rangle$ then we projected onto the $+1$ -eigenspace of ZZ , which is spanned by the vectors $|00\rangle$ and $|11\rangle$; otherwise, we projected onto the -1 -eigenspace of ZZ , which is spanned by the vectors $|01\rangle$ and $|10\rangle$. The circuit above is more commonly drawn simply as



This scheme can be generalised and used to implement any sequence of Pauli measurements. For example, the circuit below shows two consecutive measurements: $X1Z$ followed by $1YZ$.



7.8.2 The Pauli group of three qubits

Consider the three-qubit Pauli group \mathcal{P}_3 , which has $4 \cdot 4^3 = 256$ elements. One example of a stabiliser group is

$$\begin{aligned}\mathcal{S} &= \{\mathbf{111}, ZZ\mathbf{1}, \mathbf{1ZZ}, Z\mathbf{1Z}\} \\ &= \langle ZZ\mathbf{1}, \mathbf{1ZZ} \rangle.\end{aligned}$$

since it is an abelian subgroup of \mathcal{P}_3 that does not contain $-\mathbf{111}$. You can check that it fixes the subspace spanned by $|000\rangle$ and $|111\rangle$ (and note that you only need to check this on the generators of \mathcal{S} , not for all elements of \mathcal{S}). We have already seen in Figure 7.1 how these two generators bisect the Hilbert space of three qubits, so now let's try to understand Figure 7.2 for this specific example.

The elements of \mathcal{P}_3 that commute with the stabiliser \mathcal{S} form the normaliser $N(\mathcal{S})$. Since the stabiliser is abelian, it itself is contained inside the normaliser, but there are also elements in the normaliser that are *not* in the stabiliser. All together, there are $4 \cdot 16 = 64$ elements in the normaliser, and they can be neatly sliced into cosets of \mathcal{S} in $N(\mathcal{S})$, as shown in Figure 7.3.

				$-iZZZ$	$-i\mathbf{11}Z$	$-iZ\mathbf{11}$	$-i\mathbf{1}Z\mathbf{1}$	
				$iYYY$	$-iXXY$	$-iYXX$	$-iXYX$	
				$iZZZ$	$i\mathbf{11}Z$	$iZ\mathbf{11}$	$i\mathbf{1}Z\mathbf{1}$	$iYXY$
				$-iYYY$	$iXXY$	$iYXX$	$iXYX$	$-iZ\mathbf{1}Z$
				$-ZZZ$	$-\mathbf{11}Z$	$-Z\mathbf{11}$	$-\mathbf{1}Z\mathbf{1}$	Y
				YYY	$-XXY$	$-YXX$	$-XYX$	Z
ZZZ	$\mathbf{11}Z$	$Z\mathbf{11}$	$\mathbf{1}Z\mathbf{1}$			YXY		
$-YYY$	XXY	YXX	XYX			Z	$-Z\mathbf{1}Z$	
XXX	$-YYX$	$-XYY$	$-YXY$					
$\mathbf{111}$	$ZZ\mathbf{1}$	$\mathbf{1}ZZ$	$Z\mathbf{1}Z$					

Figure 7.3: Here we have arranged the elements of the normaliser of \mathcal{S} so that each row represents a coset of \mathcal{S} in $N(\mathcal{S})$. The quotient group $N(\mathcal{S})/\mathcal{S}$ is isomorphic to the Pauli group \mathcal{P}_1 , and you can see this by considering the first column (which is the **representative** for that row/coset) of the frontmost page: the four operators $\mathbf{111}$, XXX , $-YYY$, and ZZZ behave, algebraically, exactly the same as $\mathbf{1}$, X , Y , and Z (in that they satisfy the same commutation relations). Note that it is indeed $-YYY$ that behaves like Y , not $+YYY$.

Having pictured the cosets of \mathcal{S} inside $N(\mathcal{S})$, we can now look at the cosets of $N(\mathcal{S})$ inside \mathcal{P}_3 , as in Figure 7.4. A “filled in” version of this diagram (where every element is listed, as in Figure 7.3) will be given in Figure 14.8.

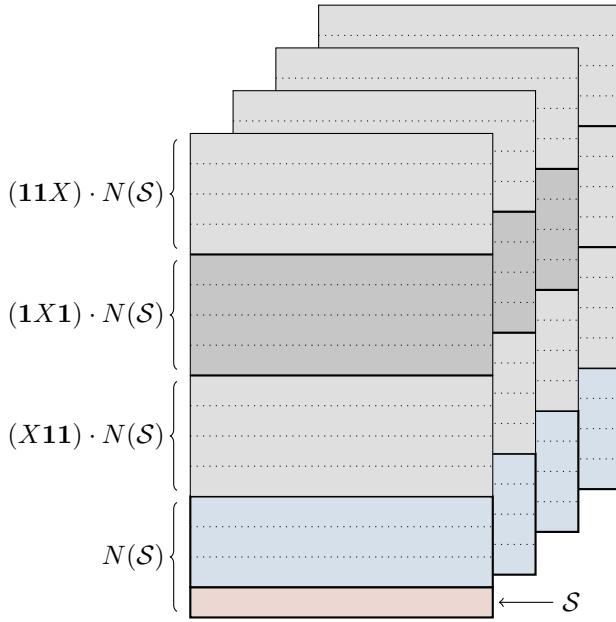


Figure 7.4: Here the cosets of the normaliser $N(\mathcal{S})$ inside \mathcal{P}_3 are given by the representatives 111 , $X11$, $1X1$, and $11X$. Each of the four cosets representing an element of $\mathcal{P}_3/N(\mathcal{S})$ is composed of 16 rows (four in each sheet). These rows represent cosets of \mathcal{S} in \mathcal{P}_3 but we have to be careful: within the normaliser $N(\mathcal{S})$, these are well defined, but outside of the normaliser there is a difference between left and right cosets, since \mathcal{S} is not normal in \mathcal{P}_3 . The blue and red rows are exactly a copy of those from Figure 7.3.

7.8.3 Half commuting

Any Pauli matrix that is not the identity commutes with exactly half of all the Pauli matrices: namely, with the identity and with itself. For example, X commutes with 1 and X , and anticommutes with Y and Z .

Extend this observation to any non-identity element in \mathcal{P}_n . In other words, show that, for any $P \in \mathcal{P}_n \setminus \{1^{\otimes n}\}$, exactly half of the elements in \mathcal{P}_n commute with P .

7.8.4 One out of four stabilisers

Explain why, if S is an element of some stabiliser group, then none of $-S$, iS , or $-iS$ are in the same stabiliser group.

7.8.5 Stabilisers and projectors

Let \mathcal{S} be a Pauli stabiliser group, with generators G_1, \dots, G_r .

1. Show that $\frac{1}{2}(1 \pm G_j)$ is the projector onto the ± 1 -eigenspace of G_j for any $j = 1, \dots, r$.
2. Show that the projector

$$P = \frac{1}{2}(1 + G_1) \frac{1}{2}(1 + G_2) \dots \frac{1}{2}(1 + G_r)$$

onto the simultaneous $+1$ -eigenspace of the generators G_1, \dots, G_r can be written as

$$P = \frac{1}{2^r} (S_1 + S_2 + \dots + S_{2^r})$$

where the sum contains all elements S_i of \mathcal{S} .

3. The fact that independent generators consecutively bisect the total Hilbert space relies on the fact that, if G_1 and G_2 are independent generators, then G_2 restricted to the $+1$ -eigenspace of G_1 bisects it into two subspaces of equal dimension. Explain how this fact follows from

$$\text{tr} \left[\frac{1}{2}(\mathbf{1} + G_1)G_2 \frac{1}{2}(\mathbf{1} + G_1) \right] = 0$$

and prove that this trace is indeed zero. Why do the two generators G_1 and G_2 have to be independent?

7.8.6 Abelian Pauli quotients

Given any group G , we define the **commutator** $[-, -]$ by

$$\begin{aligned} [-, -] : G \times G &\longrightarrow G \\ (g_1, g_2) &\longmapsto g_1^{-1}g_2^{-1}g_1g_2 \end{aligned}$$

Now let $H \triangleleft G$ be a normal subgroup, and consider the following theorem.

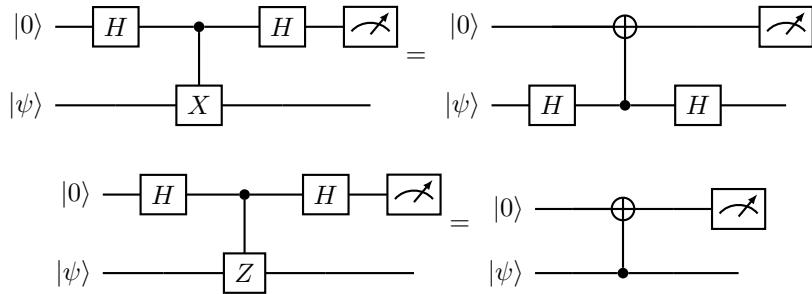
Theorem. The quotient G/H is abelian if and only if $[g_1, g_2] \in H$ for all $g_1, g_2 \in G$.

Using this theorem (or otherwise),

1. Prove that $\mathcal{P}_n/N(\mathcal{S})$ is abelian for any Pauli stabiliser $\mathcal{S} \leq \mathcal{P}_n$.
2. Prove that \mathcal{P}_n/C_4 is abelian, where $C_4 \cong \mathbb{Z}/4\mathbb{Z}$ is given by the global phase.

7.8.7 Equivalent projective measurements

In Section 7.4 we described a generic method for constructing projective measurements of Pauli observables. However, sometimes it may be easier to use equivalent, simpler constructions. For example, the following two circuit identities are often used:



These both follows from the fact that $HZH = X$, and that the control and the target of a controlled- Z gate can be chosen arbitrarily, since the gate itself is symmetric with respect to this choice: the phase-flip only happens when *both* qubits are in state $|1\rangle$.

8 Density matrices

About **density matrices**, and how they help to solve the problem introduced by entangled states, as well as how they let us talk about mixtures and subsystems. Also a first look at the **partial trace**.

We cannot always assign a definite state vector to a quantum system. It may be that the system is part of a composite system that is in an entangled state, or it may be that our knowledge of the preparation of a particular system is insufficient to determine its state — for example, someone may prepare a particle in one of the states $|\psi_1\rangle, |\psi_2\rangle, \dots, |\psi_n\rangle$, with (respective) probabilities p_1, p_2, \dots, p_n , and then give it to us without telling us which state $|\psi_k\rangle$ it's actually in. Nevertheless, in either case we are able to make statistical predictions about the outcomes of measurements performed on the system using a more general description of quantum states.

We have already mentioned that the existence of entangled states leads to an obvious question: if we cannot attribute a state vectors to an individual quantum system, then how should we describe its quantum state? In this chapter we will introduce an alternate description of quantum states that can be applied both to a composite system and to any of its subsystems. Our new mathematical tool is called a **density operator**. We will start with the density operator as a description of the mixture of quantum states, and will then discuss the partial trace, which is a unique operation that takes care of the reduction of a density operator of a composite system to density operators of its components.

8.1 Definitions

If you are an impatient, more mathematically minded person, who feels most comfortable when things are properly defined right from the beginning, here is your definition. Recall that a Hermitian matrix M is said to be **non-negative**, or **positive semi-definite**, if $\langle v|M|v\rangle \geq 0$ for any vector $|v\rangle$, or if all of its eigenvalues are non-negative, or if there exists another matrix A such that $M = A^\dagger A$.

If we choose a particular basis, operators become matrices. Throughout this book we use both terms (*density operators* and *density matrices*) pretty interchangeably.

(This is called a **Cholesky factorization**.)

A **density operator** ρ on a Hilbert space \mathcal{H} is a non-negative Hermitian operator with trace equal to one:

- **Hermitian:** $\rho^\dagger = \rho$
- **Non-negative:** $\langle v|\rho|v\rangle \geq 0$ for all $|v\rangle$
- **Trace one:** $\text{tr } \rho = 1$.

It follows that any density operator ρ can always be diagonalised, and that the eigenvalues are all real, non-negative, and sum to 1. Moreover, given two density operators ρ_1 and ρ_2 , we can always construct another density operator as a **convex sum** of the two:

$$\rho = p_1\rho_1 + p_2\rho_2$$

where $p_1, p_2 \geq 0$ are such that $p_1 + p_2 = 1$. You should check that the resulting ρ has all the defining properties of a density matrix, i.e. that it is Hermitian, non-negative, and that its trace is 1. This means that density operators form a **convex set**: a subset of a vector space is said to be **convex** if, for any two points in the subset, the straight line segment joining them is also entirely contained inside the subset.

An important example of a density operator is a rank-one projector: any quantum state that can be described by the state vector $|\psi\rangle$ can be also described by the density operator $\rho = |\psi\rangle\langle\psi|$; such states are called **pure states**. Pure states are the extremal

Note that these properties are exactly saying that we can interpret the eigenvalues as *probabilities*.

Recall that the rank of a matrix is equal to the number of its non-zero eigenvalues, or (equivalently) the dimension of its image.

points in the convex set of density operators: they cannot be expressed as a non-trivial convex sum of other elements in the set. In contrast, all other states, called **mixed states**, can be always written as the convex sum of pure states: $\sum_i p_i |\psi_i\rangle\langle\psi_i|$ for some $p_i \geq 0$ with $\sum_i p_i = 1$.

Convex spaces.

Convex spaces show up in many areas of mathematics: combinatorists and discrete geometers are often interested in **convex polytopes**, and the special case of **simplices** is even more fundamental, turning up in algebraic topology, higher algebraic geometry, and, more generally, **higher category theory**. Closer to what we are studying, the notion of **entropy** in (classical) information theory is somehow inherently convex — see e.g. Baez, Fritz, and Leinster’s “A Characterization of Entropy in Terms of Information Loss”, arXiv:[1106.1791](#).

The specific type of convex polytope that we are interested in turns out to be a **convex hull**, and these are also found all throughout mathematics.

Now that we have settled the mathematical essentials, we will turn to physical applications.

8.2 Statistical mixtures

Let us start with probability distributions over state vectors. Suppose Alice prepares a quantum system and hands it over to Bob, who subsequently measures observable M . If Alice’s preparation is described by a state vector $|\psi\rangle$, then, quantum theory declares, the average value of any observable M is given by $\langle\psi|M|\psi\rangle$, which we have previously also written as

$$\langle M \rangle = \langle \psi | M | \psi \rangle = \text{tr } M |\psi\rangle\langle\psi|.$$

This way of expressing the average value makes a clear separation between the contributions from the state preparation and from the choice of the measurement. We have two operators inside the trace: $|\psi\rangle\langle\psi|$ describes the state preparation, and M describes the measurement.

Now, suppose Alice prepares the quantum system in one of the (normalised, but not necessarily orthogonal) states $|\psi_1\rangle, \dots, |\psi_m\rangle$, choosing state $|\psi_i\rangle$ with probability p_i . She then hands the system to Bob without telling him which state she chose. We call this situation a **(statistical) mixture of the states** $|\psi_i\rangle$, or a **mixed state** for short.

It is important to note that a mixture of states is very different from a superposition of states: a superposition *always* yields a definite state vector, whereas a mixture does *not*, and so must be described by a density operator.

Let’s be extra clear about this distinction between superpositions and statistical mixtures. If Alice had prepared the system in the *superposition* $\sum_i p_i |\psi_i\rangle$, then both her *and* Bob would describe it by the state vector $\sum_i p_i |\psi_i\rangle$. If she instead follows the above random procedure, then *she* knows that it is simply described by the state vector $|\psi_i\rangle$, but the best “description” available to *Bob* is $\sum_i p_i |\psi_i\rangle\langle\psi_i|$, as we will now justify.

What Bob *does* know is the ensemble of states $|\psi_1\rangle, \dots, |\psi_m\rangle$ as well as the corresponding probability distribution p_1, \dots, p_m . Using this, he can calculate $\langle M \rangle$ as

For brevity, we often simply say “**probability distribution**” to mean “a finite set of non-negative real numbers p_k such that $\sum_k p_k = 1$ ”.

If M is one of the orthogonal projectors P_k describing the measurement, then the average $\langle P_k \rangle$ is the probability of the outcome k associated with this projector.

A pure state can be seen as a special case of a mixed state, where all but one the probabilities p_i equal zero. So by talking about mixed states, we’re still able to talk about everything that we’ve already seen up to this point.

This description is not one that we have seen before — it’s not a linear combination of kets, but instead a linear combination of *projectors*!

follows:

$$\begin{aligned}\langle M \rangle &= \sum_i p_i (\text{tr } M |\psi_i\rangle\langle\psi_i|) \\ &= \text{tr } M \left(\sum_i p_i |\psi_i\rangle\langle\psi_i| \right) \\ &= \text{tr } M \rho\end{aligned}$$

where we have simply defined $\rho = \sum_i p_i |\psi_i\rangle\langle\psi_i|$. As before, we have two operators under the trace: $\rho = \sum_i p_i |\psi_i\rangle\langle\psi_i|$, which pertains to the state preparation, and M , which describes the measurement. We shall call the operator

$$\rho = \sum_i p_i |\psi_i\rangle\langle\psi_i|$$

the **associated density operator**, since it has all the defining properties of a density operator (it is a convex sum of rank-one projectors). It depends on the constituent states $|\psi_i\rangle$ and their probabilities, and it describes our ignorance about the state preparation. Conversely, given a density operator ρ , then we call a set $\{(p_i, |\psi_i\rangle\langle\psi_i|)\}$ a **convex decomposition** if it expresses ρ as a convex sum of rank-one projectors, i.e. if $\rho = \sum_i p_i |\psi_i\rangle\langle\psi_i|$.

Once we have ρ we can make statistical predictions: we have just shown that, for any observable M , its expected value is given by

$$\langle M \rangle = \text{tr } M \rho.$$

So the exact composition of the mixture does not enter this formula: for computing the statistics associated with any observable property of a system, all that matters is the density operator itself, but *not* its decomposition into the mixture of states. This is important because any given density operator, with the remarkable exception of a pure state, can arise from many different mixtures of pure states. Consider, for example, the following three scenarios:

1. Alice flips a fair coin. If the result is heads then she prepares the qubit in the state $|0\rangle$, and if the result is tails then she prepares the qubit in the state $|1\rangle$. She gives Bob the qubit without revealing the result of the coin-flip. Bob's knowledge of the qubit is described by the density matrix

$$\frac{1}{2}|0\rangle\langle 0| + \frac{1}{2}|1\rangle\langle 1| = \begin{bmatrix} \frac{1}{2} & 0 \\ 0 & \frac{1}{2} \end{bmatrix}.$$

2. Alice flips a fair coin. If the result is heads then she prepares the qubit in the state $|+\rangle := \frac{1}{\sqrt{2}}(|0\rangle + |1\rangle)$, and if the result is tails then she prepares the qubit in the state $|-\rangle := \frac{1}{\sqrt{2}}(|0\rangle - |1\rangle)$. Bob's knowledge of the qubit is now described by the density matrix

$$\begin{aligned}\frac{1}{2}|+\rangle\langle+| + \frac{1}{2}|-\rangle\langle-| &= \frac{1}{2} \begin{bmatrix} \frac{1}{2} & \frac{1}{2} \\ \frac{1}{2} & \frac{1}{2} \end{bmatrix} + \frac{1}{2} \begin{bmatrix} \frac{1}{2} & -\frac{1}{2} \\ -\frac{1}{2} & \frac{1}{2} \end{bmatrix} \\ &= \begin{bmatrix} \frac{1}{2} & 0 \\ 0 & \frac{1}{2} \end{bmatrix}.\end{aligned}$$

3. Alice flips a fair coin, having already picked an arbitrary pair of orthonormal states $|u_1\rangle$ and $|u_2\rangle$. If the result is heads then she prepares the qubit in the state $|u_1\rangle$, and if the result is tails then she prepares the qubit in the state $|u_2\rangle$. Since any two orthonormal states of a qubit form a complete basis, the mixture $\frac{1}{2}|u_1\rangle\langle u_1| + \frac{1}{2}|u_2\rangle\langle u_2|$ gives $\frac{1}{2}\mathbf{1}$.

As you can see, these three different preparations yield precisely the same density matrix and are thus *statistically indistinguishable*. In general, two different mixtures can be distinguished (in a statistical, experimental sense) if and only if they yield different density matrices. In fact, the optimal way of distinguishing quantum states with different density operators is still an active area of research.

8.3 Instructive examples

The density matrix corresponding to the state vector $|\psi\rangle$ is the rank-one projector $|\psi\rangle\langle\psi|$.

This correspondence is well defined: each $|\psi\rangle$ gives rise to a distinct density matrix, and the fact that we ignore global phases for state vectors doesn't introduce any ambiguity for the density matrices, since $|\psi\rangle$ and $e^{i\phi}|\psi\rangle$ give the same density matrix.

Let's consider two examples, seeing again how superpositions differ from statistical mixtures.

1. If Alice prepares a qubit in the superposition state $|\psi\rangle = \alpha|0\rangle + \beta|1\rangle$ then the corresponding density matrix is the projector

$$|\psi\rangle\langle\psi| = \begin{bmatrix} |\alpha|^2 & \alpha\beta^* \\ \alpha^*\beta & |\beta|^2 \end{bmatrix}.$$

2. You are given a qubit and you are told that it was prepared either in state $|0\rangle$ with probability $|\alpha|^2$ or in state $|1\rangle$ with probability $|\beta|^2$. In this case all you can say is that your qubit is in a mixed state described by the density matrix

$$|\alpha|^2|0\rangle\langle 0| + |\beta|^2|1\rangle\langle 1| = \begin{bmatrix} |\alpha|^2 & 0 \\ 0 & |\beta|^2 \end{bmatrix}.$$

The density matrix corresponding to a statistical mixture of states $|\psi_1\rangle, \dots, |\psi_n\rangle$ with probability distribution p_1, \dots, p_n is the convex combination $\sum_i p_i |\psi_i\rangle\langle\psi_i|$. If the constituent states are orthogonal, then the density matrix is diagonal.

Suppose you want to distinguish between preparations described by the density matrices in the above two examples. Assume that you are given sufficiently many qubits, all identically prepared, i.e. either all described by the density matrix $\begin{bmatrix} |\alpha|^2 & \alpha\beta^* \\ \alpha^*\beta & |\beta|^2 \end{bmatrix}$, or all described by the density matrix $\begin{bmatrix} |\alpha|^2 & 0 \\ 0 & |\beta|^2 \end{bmatrix}$. Which of the two measurements would you choose: the measurement in the standard basis $\{|0\rangle, |1\rangle\}$, or the measurement in the basis $\{|\psi\rangle, |\psi^\perp\rangle\}$ where $|\psi^\perp\rangle$ is orthonormal to $|\psi\rangle$?

In general, the diagonal entries of a density matrix describe the probability distributions on the set of basis vectors. They must add up to one, which is why the trace of any density matrix is one. The off-diagonal elements, often called **coherences**, signal departure from the classical probability distribution and quantify the degree to which a quantum system can witness interference (we will discuss this in detail later on). The process in which off-diagonal entries go to zero is called **decoherence**.

$$\begin{bmatrix} |\alpha|^2 & \alpha\beta^* \\ \alpha^*\beta & |\beta|^2 \end{bmatrix} \mapsto \begin{bmatrix} |\alpha|^2 & \varepsilon \\ \varepsilon^* & |\beta|^2 \end{bmatrix} \mapsto \begin{bmatrix} |\alpha|^2 & 0 \\ 0 & |\beta|^2 \end{bmatrix}$$

For $\varepsilon = \alpha\beta^*$ we have a pure quantum state ("full interference capability") and for $\varepsilon = 0$ we have a classical probability distribution over the standard basis ("no interference capability").

In fact, one of these two measurements is completely useless.
Exercise. Which one, and why?

3. Suppose that your qubit was prepared either in state $\alpha|0\rangle + \beta|1\rangle$ or in state $\alpha|0\rangle - \beta|1\rangle$, with equal probability. This means that your qubit is in a mixed state described by the density matrix

$$\frac{1}{2} \begin{bmatrix} |\alpha|^2 & \alpha\beta^* \\ \alpha^*\beta & |\beta|^2 \end{bmatrix} + \frac{1}{2} \begin{bmatrix} |\alpha|^2 & -\alpha\beta^* \\ -\alpha^*\beta & |\beta|^2 \end{bmatrix} = \begin{bmatrix} |\alpha|^2 & 0 \\ 0 & |\beta|^2 \end{bmatrix}.$$

There is *no way* to tell the difference between the equally weighted mixture of $\alpha|0\rangle \pm \beta|1\rangle$ and a mixture of $|0\rangle$ and $|1\rangle$ with (respective) probabilities $|\alpha|^2$ and $|\beta|^2$.

4. For *any* density matrix ρ , the most natural mixture that yields ρ is its **spectral decomposition**: $\rho = \sum_i p_i |u_i\rangle\langle u_i|$, with eigenvectors $|u_i\rangle$ and eigenvalues p_i .
5. If the states $|u_1\rangle, \dots, |u_n\rangle$ form an orthonormal basis, and each occurs with equal probability $1/n$, then the resulting density matrix is proportional to the identity:

$$\frac{1}{n} \sum_{i=1}^n |\psi_i\rangle\langle\psi_i| = \frac{1}{n} \mathbf{1}.$$

This is a **maximally mixed** state. For qubits, any pair of orthogonal states taken with equal probabilities gives the maximally mixed state $\frac{1}{2}\mathbf{1}$.

A state is said to be **maximally mixed** if the outcomes of *any* measurement are completely random.

It is often convenient to write density operators in terms of projectors on states which are *not* normalised, incorporating the probabilities into the length of the state vector:

$$\rho = \sum_i |\tilde{\psi}_i\rangle\langle\tilde{\psi}_i|$$

where $|\tilde{\psi}_i\rangle = \sqrt{p_i}|\psi_i\rangle$, i.e. $p_i = \langle\tilde{\psi}_i|\tilde{\psi}_i\rangle$. This form is more compact, but you have to remember that the state vectors are *not* normalised. We tend to mark such states with the tilde, e.g. $|\tilde{\psi}\rangle$, but you may have your own way to remember.

8.4 The Bloch ball

We have already talked in some depth about the Bloch sphere, but now that we are considering density operators (which are strictly more general than state vectors), we are actually interested in the Bloch *ball*, i.e. not just the sphere of vectors of magnitude 1, but instead the ball of vectors of magnitude *less than or equal* to 1.

An arbitrary (2×2) Hermitian matrix has four real parameters and can be expanded in the basis $\{\mathbf{1}, \sigma_x, \sigma_y, \sigma_z\}$ consisting of the identity and the three Pauli matrices. Since the Pauli matrices are traceless (i.e. their trace is equal to 0), the coefficient of $\mathbf{1}$ in the expansion of a density matrix ρ must be $\frac{1}{2}$, in order to have $\text{tr } \rho = 1$. Thus ρ may be expressed as

$$\begin{aligned} \rho &= \frac{1}{2} (\mathbf{1} + \vec{s} \cdot \vec{\sigma}) \\ &= \frac{1}{2} \begin{bmatrix} 1 + s_z & s_x - is_y \\ s_x + is_y & 1 - s_z \end{bmatrix}. \end{aligned}$$

Physicists often still refer to the Bloch *ball* as the Bloch *sphere*, even though it really is a ball now, not a sphere.

where $\vec{s} = (s_x, s_y, s_z)$ and $\vec{\sigma} = (\sigma_x, \sigma_y, \sigma_z)$. The vector \vec{s} is called the **Bloch vector** for the density operator ρ . Any real Bloch vector \vec{s} defines a Hermitian operator ρ with $\text{tr } \rho = 1$, but in order for ρ to be a density operator it must also be non-negative. Which Bloch vectors yield legitimate density operators? That is, what does the non-negative condition on ρ translate to in terms of the Bloch vector \vec{s} ?

To answer this, let us compute the eigenvalues of ρ . The trace of a matrix is equal to the *sum* of its eigenvalues, and the determinant is equal to the *product* of its eigenvalues. We know that $\text{tr } \rho = 1$, and we can calculate $\det \rho$ from the matrix form above:

$$\begin{aligned}\det \rho &= \frac{1}{4}(1 - s^2) \\ &= \frac{1}{2}(1 + s)\frac{1}{2}(1 - s)\end{aligned}$$

where $s = |\vec{s}| = \sqrt{|s_x|^2 + |s_y|^2 + |s_z|^2}$. It follows that the two eigenvalues of ρ are $\frac{1}{2}(1 \pm s)$. For ρ to be non-negative, its eigenvalues have to be non-negative, and so s (the length of the Bloch vector) cannot exceed 1.

We can now visualise the convex set of (2×2) density matrices as a unit ball in three-dimensional Euclidean space: the extremal points, which represent pure states, are the points on the boundary (\vec{s} such that $s = 1$), i.e. the surface of the ball (the Bloch sphere, which we have already seen!); the maximally mixed state $1/2$ corresponds to $s = 0$, i.e. the centre of the ball. In general, the length of the Bloch vector s can be thought of as the “purity” of a state.

One might hope that there is an equally simple visualisation of the density operators in higher dimensions. Unfortunately, there is *not*: things become *much* more complicated, very quickly.

Bloch ball for qutrits.

Qubits are 2-dimensional and give rise to the Bloch ball, which is a 3-dimensional object. In general, n -dimensional quantum systems give rise to $(n^2 - 1)$ -dimensional state spaces, often denoted \mathcal{Q}_n ; for $n = 3$, where we study **qutrits**, we would need to study an 8-dimensional object \mathcal{Q}_3 .

It turns out, quite surprisingly, that there exists a 3-dimensional object that has many (but not all) of the properties that we would want from \mathcal{Q}_3 . For example, the rank-1 pure states form a connected set on the surface, which lies a maximum distance of $\sqrt{2}$ from the maximally mixed state $\frac{1}{3}\mathbf{1}$; the other points on the surface correspond to rank-1 and rank-2 operators; the points strictly inside correspond to rank-3 (i.e. full rank) operators. However, since it is only 3-dimensional, it can never satisfy *all* the properties that we would like, since \mathcal{Q}_3 *has to be* 8-dimensional. Nevertheless, the construction is both interesting and useful (and very recent!) — see C Eltschka, M Huber, S Morelli, and J Siewert, “The shape of higher-dimensional state space: Bloch-ball analog for a qutrit”, *Quantum* 5 (2021), DOI: [10.22331/q-2021-06-29-485](https://doi.org/10.22331/q-2021-06-29-485).

One has to be careful when trying to use the Bloch ball to talk about multiple qubits, precisely for the reason that “most” states are not separable states, but instead have some amount of entanglement. If we have n qubits, then we can describe the corresponding product state in terms of n vectors in the Bloch ball, but this method only lets us describe *product* states of the n qubits — we saw in Section 5.5 that, as n grows larger, “most” states are *not* separable!

For example, say that we have a system with two qubits, and we wish to understand how they move around the Bloch sphere under some unitary evolution. If our qubits are initially in state $|a\rangle|b\rangle$, then evolve to the state $U|a\rangle|b\rangle$. Simple! But now say that, before applying our unitary U , we first *rotated* the Bloch ball so that our

qubits were in some other state $|a'\rangle|b'\rangle$, and *then* applied our unitary U to this rotated state. A natural question to ask is if there exists some rotation that takes the first result $U|a\rangle|b\rangle$ to the second result $U|a'\rangle|b'\rangle$. In other words, if we denote our rotation by R , then does there exist a rotation S such that $U \circ R = S \circ U$?

The answer is most definitely *no*, as shown by a reasonably simple example: consider the controlled-NOT gate acting on two qubits initially in some state $|0\rangle|\psi\rangle$, and where the rotation R takes $|0\rangle|\psi\rangle$ to $|\psi'\rangle|0\rangle$. Then $(U \circ R)|a\rangle|b\rangle = |\psi'\rangle|\psi'\rangle$, and $U|a\rangle|b\rangle = |0\rangle|\psi\rangle$. But we cannot transform the latter into the former by a simple rotation of the sphere, since the latter has two distinct Bloch vectors, whereas the former has a single repeated one, and rotations never “collapse” two distinct vectors into one. The key point here is that the angles between the Bloch vectors can *change* upon applying unitary operations, and the amount by which they change can depend on the Bloch vectors themselves, whereas rotations keep these relative angles *constant*.

8.5 Subsystems of entangled systems

Earlier, we claimed that one of the most important features of the density operator formalism is its ability to describe the quantum state of a subsystem of a composite system. Let us now show you how this works.

Given a quantum state of the composite system \mathcal{AB} described by some density operator $\rho_{\mathcal{AB}}$, we obtain **reduced** density operators ρ_A and ρ_B of the subsystems \mathcal{A} and \mathcal{B} (respectively) by the **partial trace**:

$$\begin{aligned}\rho_{\mathcal{AB}} \longmapsto \rho_A &= \underbrace{\text{tr}_B \rho_{\mathcal{AB}}}_{\text{partial trace over } \mathcal{B}} \\ \rho_{\mathcal{AB}} \longmapsto \rho_B &= \underbrace{\text{tr}_A \rho_{\mathcal{AB}}}_{\text{partial trace over } \mathcal{A}}\end{aligned}$$

We will revisit the notion of partial trace quite a few times, but for now we simply define the partial trace over \mathcal{B} (or \mathcal{A}) first on a tensor product of two operators $A \otimes B$ as

$$\begin{aligned}\text{tr}_B(A \otimes B) &= A(\text{tr} B) \\ \text{tr}_A(A \otimes B) &= (\text{tr} A)B,\end{aligned}$$

and then extend to any operator on $\mathcal{H}_A \otimes \mathcal{H}_B$ by linearity.

Here is a simple example. Suppose a composite system \mathcal{AB} is in a pure entangled state $|\psi_{\mathcal{AB}}\rangle$. We can always write this as

$$|\psi_{\mathcal{AB}}\rangle = \sum_i c_i |a_i\rangle \otimes |b_i\rangle,$$

where $|a_i\rangle$ and $|b_j\rangle$ are two orthonormal bases (e.g. the Schmidt bases, from Exercise 5.14.13), and where $\sum_i |c_i|^2 = 1$ (due to the normalisation). The corresponding density operator of the composite system is the projector $\rho_{\mathcal{AB}} = |\psi_{\mathcal{AB}}\rangle\langle\psi_{\mathcal{AB}}|$, which we can write as

$$\rho_{\mathcal{AB}} = |\psi_{\mathcal{AB}}\rangle\langle\psi_{\mathcal{AB}}| = \sum_{i,j} c_i c_j^* |a_i\rangle\langle a_j| \otimes |b_i\rangle\langle b_j|$$

Let us compute the reduced density operator ρ_A by taking the partial trace over

\mathcal{B} :

$$\begin{aligned}
 \rho_{\mathcal{A}} &= \text{tr}_{\mathcal{B}} \rho_{\mathcal{AB}} \\
 &= \text{tr}_{\mathcal{B}} |\psi_{\mathcal{AB}}\rangle\langle\psi_{\mathcal{AB}}| \\
 &= \text{tr}_{\mathcal{B}} \sum_{i,j} c_i c_j^* |a_i\rangle\langle a_j| \otimes |b_i\rangle\langle b_j| \\
 &= \sum_{i,j} c_i c_j^* |a_i\rangle\langle a_j| (\text{tr} |b_i\rangle\langle b_j|) \\
 &= \sum_{i,j} c_i c_j^* |a_i\rangle\langle a_j| \underbrace{\langle b_i|b_j\rangle}_{\delta_{ij}} \\
 &= \sum_i |c_i|^2 |a_i\rangle\langle a_i|.
 \end{aligned}$$

So, in the $|a_i\rangle$ basis, the reduced density matrix $\rho_{\mathcal{A}}$ is diagonal, with entries $p_i = |c_i|^2$. Similarly, if we take the partial trace over \mathcal{A} , then we get $\rho_{\mathcal{B}} = \sum_i |c_i|^2 |b_i\rangle\langle b_i|$.

In particular, if $\dim \mathcal{H}_{\mathcal{A}} = \dim \mathcal{H}_{\mathcal{B}} = d$, then the maximally mixed state

$$|\psi_{\mathcal{AB}}\rangle = \frac{1}{\sqrt{d}} \sum_i^d |a_i\rangle |b_i\rangle,$$

in the $(d \times d)$ -dimensional Hilbert space $\mathcal{H}_{\mathcal{A}} \otimes \mathcal{H}_{\mathcal{B}}$ is such that the reduced density operators $\rho_{\mathcal{A}}$ and $\rho_{\mathcal{B}}$ are also the maximally mixed states of their respective subsystems: $\rho_{\mathcal{A}} = \rho_{\mathcal{B}} = \frac{1}{d}\mathbf{1}$. It follows that the quantum states of individual qubits in any of the Bell states are maximally mixed: their density matrix is $\frac{1}{2}\mathbf{1}$.

A bipartite state such as

$$\frac{1}{\sqrt{2}} (|00\rangle + |11\rangle)$$

guarantees *perfect correlations* when each qubit is measured in the standard basis: the two outcomes are “0 and 0” or “1 and 1” (which are equally likely), and we will never observe e.g. “0 and 1”, but the outcome of either *single-qubit* subsystem is completely random.

8.6 Mixtures and subsystems

So far we have used density operators to describe two distinct situations: the statistical properties of mixtures of states, and the statistical properties of subsystems of composite systems. In order to see the relationship between the two, consider a joint state of a bipartite system \mathcal{AB} , written in a product basis of $\mathcal{H}_{\mathcal{A}} \otimes \mathcal{H}_{\mathcal{B}}$ as

$$\begin{aligned}
 |\psi_{\mathcal{AB}}\rangle &= \sum_{i,j} c_{ij} |a_i\rangle \otimes |b_j\rangle \\
 &= \sum_j |\tilde{\psi}_j\rangle |b_j\rangle = \sum_j \sqrt{p_j} |\psi_j\rangle |b_j\rangle
 \end{aligned}$$

where $|\tilde{\psi}_j\rangle = \sum_i c_{ij} |a_i\rangle$, which we can also write as $\sqrt{p_j} |\psi_j\rangle$ where the $|\psi_j\rangle$ are the normalised versions of the $|\tilde{\psi}_j\rangle$, and $p_j = \langle \tilde{\psi}_j | \tilde{\psi}_j \rangle$.

Then the partial trace over \mathcal{B} gives the reduced density operator of subsystem \mathcal{A} :

$$\begin{aligned}\rho_{\mathcal{A}} &= \text{tr}_{\mathcal{B}} \left(\sum_{i,j} |\tilde{\psi}_i\rangle\langle\tilde{\psi}_j| \otimes |b_i\rangle\langle b_j| \right) \\ &= \sum_{i,j} |\tilde{\psi}_i\rangle\langle\tilde{\psi}_j| (\text{tr} |b_i\rangle\langle b_j|) \\ &= \sum_{i,j} |\tilde{\psi}_i\rangle\langle\tilde{\psi}_j| \langle b_j| b_i \rangle \\ &= \sum_i |\tilde{\psi}_i\rangle\langle\tilde{\psi}_i| = \sum_i p_i |\psi_i\rangle\langle\psi_i|.\end{aligned}$$

Now let us see how $\rho_{\mathcal{A}}$ can be understood in terms of mixtures. Imagine we place subsystems \mathcal{A} and \mathcal{B} in two separate labs, run by Alice and Bob, respectively. Say Bob measures the \mathcal{B} part in the $|b_j\rangle$ basis and obtains result k , which happens with probability p_k . In doing so, he inevitably prepares subsystem \mathcal{A} in the state $|\psi_k\rangle$:

$$\sum_{i=1} \sqrt{p_i} |\psi_i\rangle |b_i\rangle \xrightarrow{\text{outcome } k} |\psi_k\rangle |b_k\rangle.$$

Bob does not communicate the outcome of his measurement. Thus, from Alice's perspective, Bob prepares a mixture of $|\psi_1\rangle, \dots, |\psi_m\rangle$, with probabilities p_1, \dots, p_m , which means that Alice, who knows the joint state but *not* the outcomes of Bob's measurement, may associate density matrix $\rho_{\mathcal{A}} = \sum_i p_i |\psi_i\rangle\langle\psi_i|$ with her subsystem \mathcal{A} . This is the same $\rho_{\mathcal{A}}$ that we obtained before by taking the partial trace.

But suppose Bob chooses to measure his subsystem in some other basis. Will it have any impact on Alice's statistical predictions? Measurement in the new basis will result in a different mixture, but Alice's density operator *will not change*.

Say Bob chooses some basis $|d_i\rangle$ for his measurement. Any two orthonormal bases are connected by some unitary transformation, and so we can write $|b_i\rangle = U|d_i\rangle$ for some unitary U . The joint state can now be expressed as

$$\begin{aligned}|\psi_{AB}\rangle &= \sum_i |\tilde{\psi}_i\rangle |b_i\rangle \\ &= \sum_i |\tilde{\psi}_i\rangle \left(\sum_j U_{ij} |d_j\rangle \right) \\ &= \sum_j \underbrace{\left(\sum_i U_{ij} |\tilde{\psi}_i\rangle \right)}_{|\tilde{\phi}_j\rangle} |d_j\rangle \\ &= \sum_j |\tilde{\phi}_j\rangle |d_j\rangle.\end{aligned}$$

In terms of components, $|b_i\rangle = \sum_j U_{ij} |d_j\rangle$

If Bob measures in the $|d_i\rangle$ basis then he generates a new mixture of states $|\phi_1\rangle, \dots, |\phi_m\rangle$, which are the normalised versions of $|\tilde{\phi}_1\rangle, \dots, |\tilde{\phi}_m\rangle$, with each $|\phi_k\rangle$ occurring with probability $p_k = \langle\tilde{\phi}_k|\tilde{\phi}_k\rangle$. But this new mixture has exactly the same density operator as the previous one:

$$\begin{aligned}\sum_j |\tilde{\phi}_j\rangle\langle\tilde{\phi}_j| &= \sum_{i,j,l} U_{ij} |\tilde{\psi}_i\rangle\langle\tilde{\psi}_l| U_{lj}^* \\ &= \sum_{i,l} \underbrace{\left(\sum_j U_{ij} U_{lj}^* \right)}_{\delta_{il}} |\tilde{\psi}_i\rangle\langle\tilde{\psi}_l| \\ &= \sum_i |\tilde{\psi}_i\rangle\langle\tilde{\psi}_i|\end{aligned}$$

where we use the fact that the U_{ij} are the entries of a unitary matrix, and so $\sum_k U_{ik} U_{jk}^* = \delta_{ij}$. But this is exactly ρ_A ! So does it really matter whether Bob actually performs the measurement or not?

No — it does not.

After all, Alice and Bob may be many many miles away from each other, and if any of Bob's actions were to result in something that is physically detectable at Alice's lab, then this would amount to *instantaneous communication* between the two of them.

From the operational point of view it does not really matter whether the density operator represents our ignorance of the actual state (mixtures) or provides the only description we can have after discarding one part of an entangled state (partial trace). In the former case, the system is in some definite pure state but we do not know which. In contrast, when the density operator arises from tracing out irrelevant, or unavailable, degrees of freedom, the individual system cannot be thought to be in some definite state of which we are ignorant. Philosophy aside, the fact that the two interpretations give exactly the same predictions is useful: switching back and forth between the two pictures often offers additional insights and may even simplify lengthy calculations.

8.7 Partial trace, revisited

You can calculate the trace of a matrix by summing its diagonal entries. Can you do something similar to calculate the partial trace of a density matrix? Suppose someone writes down for you a density matrix of two qubits in the standard basis, $\{|00\rangle, |01\rangle, |10\rangle, |11\rangle\}$, and asks you to find the reduced density matrices of the individual qubits. The tensor product structure of this (4×4) matrix means that it is has a block form:

$$\rho_{AB} = \left[\begin{array}{c|c} P & Q \\ \hline R & S \end{array} \right]$$

where P, Q, R, S are (2×2) sized sub-matrices.

The two partial traces can then be evaluated as

$$\rho_A = \text{tr}_B \rho_{AB} = \left[\begin{array}{c|c} \text{tr } P & \text{tr } Q \\ \hline \text{tr } R & \text{tr } S \end{array} \right]$$

$$\rho_B = \text{tr}_A \rho_{AB} = P + S.$$

In general, for any matrix ρ in $\mathcal{H}_A \otimes \mathcal{H}_B$ that is written *in the tensor product basis*, the partial trace over A is the sum of the diagonal block matrices, and the partial trace over B is the matrix in which the block sub-matrices are replaced by their traces — see Figure 8.1.

$$\begin{array}{c} \left[\begin{array}{c|c|c|c} \cdot & \cdot & \cdot & \cdot \\ \cdot & \cdot & \cdot & \cdot \\ \hline \cdot & \cdot & \cdot & \cdot \\ \cdot & \cdot & \cdot & \cdot \\ \hline \cdot & \cdot & \cdot & \cdot \\ \cdot & \cdot & \cdot & \cdot \end{array} \right] \quad \left[\begin{array}{c|c|c|c} \cdot & \cdot & \cdot & \cdot \\ \cdot & \cdot & \cdot & \cdot \\ \hline \cdot & \cdot & \cdot & \cdot \\ \cdot & \cdot & \cdot & \cdot \\ \hline \cdot & \cdot & \cdot & \cdot \\ \cdot & \cdot & \cdot & \cdot \end{array} \right] \\ \text{tr}_A \rho \qquad \qquad \qquad \text{tr}_B \rho \end{array}$$

Figure 8.1: Visualising the two partial traces of a matrix *written in the tensor product basis*.

To better understand the partial trace, it helps to give a more abstract definition. It turns out that the partial trace over B can be defined as the unique map $\rho_{AB} \mapsto \rho_A$ such that

The two interpretations of density operators have filled volumes of academic journals. The terms **proper mixtures** and **improper mixtures** are used, mostly by philosophers, to describe the statistical mixture and the partial trace approach, respectively.

Take any of the Bell states, write its (4×4) -density matrix explicitly, and then trace over each qubit. In each case you should get the maximally mixed state.

One can repeat the same argument for the partial trace over A : it is the unique map $\rho_{AB} \mapsto \rho_B$ such that ρ_B satisfies $\text{tr}[Y\rho_B] = \text{tr}[(1 \otimes Y)\rho_{AB}]$ for any observable Y on B .

$$\text{tr}[X\rho_{\mathcal{A}}] = \text{tr}[(X \otimes \mathbf{1})\rho_{\mathcal{AB}}] \quad (\circledast)$$

holds for any observable X acting on \mathcal{A} , where $\mathbf{1}$ is the identity operator acting on \mathcal{B} . This condition ensures the consistency of statistical predictions: any observable X on \mathcal{A} can be viewed as an observable $X \otimes \mathbf{1}$ on the composite system \mathcal{AB} ; when constructing $\rho_{\mathcal{A}}$, we had better make sure that for any observable X the average value of X in the state $\rho_{\mathcal{A}}$ is the same as the average value of $X \otimes \mathbf{1}$ in the state $\rho_{\mathcal{AB}}$. This is exactly what the condition in (\circledast) guarantees.

To show that our more ad-hoc definition of the partial trace agrees with this slightly more abstract one, consider again some state $|\psi_{\mathcal{AB}}\rangle$ written in the form

$$\begin{aligned} |\psi_{\mathcal{AB}}\rangle &= \sum_{i,j} c_{ij}|a_i\rangle \otimes |b_j\rangle \\ &= \sum_j |\tilde{\psi}_j\rangle |b_j\rangle = \sum_j \sqrt{p_j} |\psi_j\rangle |b_j\rangle. \end{aligned}$$

Now assume that Alice measures some observable X on her part of the system. Such an observable can be thought of as $X \otimes \mathbf{1}$, acting on the entire system. The expected value of *this* observable in the state $|\psi_{\mathcal{AB}}\rangle$ is, by definition, $\text{tr}(X \otimes \mathbf{1})|\psi_{\mathcal{AB}}\rangle\langle\psi_{\mathcal{AB}}|$, and

$$\begin{aligned} \text{tr}[(X \otimes \mathbf{1})\rho_{\mathcal{AB}}] &= \text{tr} \left[(X \otimes \mathbf{1}) \left(\sum_{i,j} |\tilde{\psi}_i\rangle\langle\tilde{\psi}_j| \otimes |b_i\rangle\langle b_j| \right) \right] \\ &= \sum_{i,j} \left[\text{tr} (X|\tilde{\psi}_i\rangle\langle\tilde{\psi}_j|) \right] \underbrace{[\text{tr}(|b_i\rangle\langle b_j|)]}_{\delta_{ij}} \\ &= \sum_i \text{tr} [X|\tilde{\psi}_i\rangle\langle\tilde{\psi}_i|] \\ &= \text{tr} \left[X \underbrace{\sum_i p_i |\psi_i\rangle\langle\psi_i|}_{\rho_{\mathcal{A}} = \text{tr}_{\mathcal{B}} \rho_{\mathcal{AB}}} \right] \\ &= \text{tr}[X\rho_{\mathcal{A}}] \end{aligned}$$

as required.

We can also quickly prove why the partial trace is the *unique* map satisfying the condition (\circledast) . Suppose that we had some arbitrary map T satisfying this condition, i.e. such that

$$\text{tr}[XT(\rho_{\mathcal{AB}})] = \text{tr}[(X \otimes \mathbf{1})\rho_{\mathcal{AB}}]$$

for all density matrices $\rho_{\mathcal{AB}}$ and for all observables X acting on \mathcal{A} . Now, take some orthonormal (with respect to the Hilbert–Schmidt inner product $(A|B) = \frac{1}{2} \text{tr} A^\dagger B$) basis $\{M_i\}$ of the space of Hermitian matrices. Since the M_i are Hermitian, the inner product $(M_i|T(\rho_{\mathcal{AB}}))$ is just $\text{tr}[M_i T(\rho_{\mathcal{AB}})]$.

So when we expand $T(\rho_{\mathcal{AB}})$ in this basis we get

$$\begin{aligned} T(\rho_{\mathcal{AB}}) &= \sum_i (M_i|T(\rho_{\mathcal{AB}})) M_i \\ &= \sum_i \text{tr}[M_i T(\rho_{\mathcal{AB}})] M_i. \end{aligned}$$

But now we can substitute in the condition that T satisfies, giving

$$T(\rho_{\mathcal{AB}}) = \sum_i \text{tr}[(M_i \otimes \mathbf{1})\rho_{\mathcal{AB}}] M_i.$$

And we're done! Indeed, if we had started with some other such map T' then we would have arrived at the same expression, which is independent of our choice of T or T' , whence $T = T'$.

We ignore the normalisation factor of $\frac{1}{2}$ in the Hilbert–Schmidt inner product here.

Expanding an operator in a basis might seem confusing at first, but this is really just the fact that (avoiding bra-ket notation for clarity) any vector v in an inner product space with orthonormal basis $\{e_i\}$ can be expanded as $v = \sum_i \langle e_i, v \rangle e_i$, just applied to the specific case of a vector space of matrices, with the Hilbert–Schmidt inner product.

8.8 Remarks and exercises

8.8.1 Some density operator calculations

Consider two qubits in the state

$$|\psi\rangle = \frac{1}{\sqrt{2}} \left(|0\rangle \otimes \left(\sqrt{\frac{2}{3}}|0\rangle - \sqrt{\frac{1}{3}}|1\rangle \right) + |1\rangle \otimes \left(\sqrt{\frac{2}{3}}|0\rangle + \sqrt{\frac{1}{3}}|1\rangle \right) \right).$$

1. What is the density operator ρ of the two qubits corresponding to the state $|\psi\rangle$? Write it in Dirac notation, and then explicitly as a matrix in the computational basis $\{|00\rangle, |01\rangle, |10\rangle, |11\rangle\}$.
2. Find the reduced density operators ρ_1 and ρ_2 of the first and second qubit, respectively. Again, write them in both Dirac notation as well as explicitly as a matrix in the computational basis.

8.8.2 Purification of mixed states

Given a mixed state ρ , a **purification** of ρ is a pure state $|\psi\rangle\langle\psi|$ of some potentially larger system such that ρ is equal to a partial trace of $|\psi\rangle\langle\psi|$.

1. Show that an arbitrary mixed state ρ always has a purification.
2. Show that purification is unique up to unitary equivalence.
3. Let $|\psi_1\rangle$ and $|\psi_2\rangle$ in $\mathcal{H}_A \otimes \mathcal{H}_B$ be two pure states such that $\text{tr}_B |\psi_1\rangle\langle\psi_1| = \text{tr}_B |\psi_2\rangle\langle\psi_2|$. Show that $|\psi_1\rangle = \mathbf{1} \otimes U|\psi_2\rangle$ for some unitary operator U on \mathcal{H}_B .

Well done — you have just proved the **Schrödinger–HJW theorem!**

8.8.3 Pure partial trace

Two qubits are in the state described by the density operator $\rho = \rho_A \otimes \rho_B$. What is the partial trace of ρ over each qubit?

8.8.4 Maximally Bell

What is the density matrix corresponding to two qubits prepared in the mixture of the Bell state $\Phi^+ = \frac{1}{\sqrt{2}}(|00\rangle + |11\rangle)$ and the maximally mixed state, both with equal probability $\frac{1}{2}$?

The maximally mixed state of two qubits is described by a (4×4) matrix in $\mathcal{H}_A \otimes \mathcal{H}_B$.

8.8.5 Spectral decompositions and common eigenbases

This section is not yet finished.

9 Quantum channels

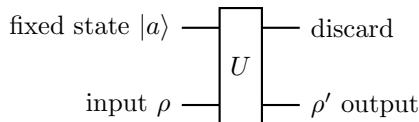
About **quantum channels**, which are to density operators what unitaries are to state vectors: mathematical models for physically realisable transformations. Also about many eponymous constructions, such as the **Stinespring** and the **Kraus representations**, the **Jamiolkowski isomorphism**, and the **Choi matrix**.

Quantum evolution of any *isolated* system is unitary, but its constituent parts may evolve in a more complicated way.

We have already discussed how entanglement forces us to describe quantum states of *open* quantum systems (i.e. those which are only part of some larger system) in terms of density operators. In this chapter we will describe how open systems evolve. The question we are asking here is: what are the most general physically admissible transformations of density operators? That is, if state vectors evolve according to unitary operations, and we generalise state vectors to density operators, then what is the “good” corresponding generalisation of unitary operations?

9.1 Everything is (secretly) unitary

At the fundamental level — and this should be your quantum mantra — there is *only* unitary evolution, and if there is any other evolution then it has to be derived from a unitary evolution. From this perspective, any *non-unitary* evolution of an open system is induced by a unitary evolution of a larger system — all evolutions become unitary when you make your system large enough! But how? The short answer is: by adding (via *tensoring*) and removing (via *partial trace*) physical systems. A typical combination of these operations is shown in the following diagram:



Let’s explain what this diagram is really saying.

First, as always, we prepare our system of interest in an input state ρ .

Next we dilate the system by “**adding**” (or “**taking into account**”) an auxiliary system which is large enough to include everything our system will interact with, and also large enough to be in a pure state $|a\rangle$. Mathematically, we do this by tensoring the input state ρ with $|a\rangle\langle a|$ to obtain $|a\rangle\langle a| \otimes \rho$ (here we place the auxiliary system first and our system of interest second). Importantly, we assume that we have “added in” a large enough auxiliary system that the resulting dilated system is *closed*, and thus undergoes *unitary* evolution, described by some U , resulting in the state $U(|a\rangle\langle a| \otimes \rho)U^\dagger$.

Finally, after all the (unitary) interactions have taken place, we **trace out** the auxiliary system, turning the joint state $U(|a\rangle\langle a| \otimes \rho)U^\dagger$ of the dilated system into the final state of our system of interest: the output state ρ' .

We shall later show that the net effect of these three operations (adding, unitary evolution, and tracing out) can be written, as long as the initial state $|a\rangle$ of the auxiliary system is *not* correlated with the input state ρ , in a nice compact way:

$$\rho \longrightarrow \rho' = \sum_i E_i \rho E_i^\dagger$$

... There is only unitary evolution. There is only unitary evolution. There is only unitary evolution... ...and everything else is cheating.

Depending on the context, the auxiliary system is either called the **ancilla** (usually when we can control it) or the **environment** (usually when we cannot control it).

where the E_i are some operators that satisfy $\sum_i E_i^\dagger E_i = \mathbf{1}$. Such a linear map is called a **completely positive trace-preserving map**, or, in the parlance of quantum information science, a **quantum channel**.

We will elaborate on the mathematics behind quantum channels shortly, but for now let us only check the essential properties, i.e. that this map preserves both trace and positivity (as its name suggests).

- *Trace preserving.* Since the trace is linear, invariant under cyclic permutations of operators, and we ask that $\sum_i E_i^\dagger E_i = \mathbf{1}$, we see that

$$\mathrm{tr} \left(\sum_k E_k \rho E_k^\dagger \right) = \mathrm{tr} \left(\sum_k E_k^\dagger E_k \rho \right) = \mathrm{tr} \rho.$$

- *Positivity preserving.* Since ρ is a positive (semi-definite) operator, so too is $\sqrt{\rho}$, and we thus see that

$$\sum_k E_k \rho E_k^\dagger = \sum_k (E_k \sqrt{\rho})(\sqrt{\rho} E_k^\dagger).$$

Recall that an operator is positive if and only if it can be written in the form XX^\dagger for some X (here $X = E_k \sqrt{\rho}$). Also, the sum of positive operators is again a positive operator.

These conditions are certainly *necessary* if we want to map density operators into legal density operators, but we shall see in a moment that they are *not sufficient*: quantum channels are not just positive maps, but instead **completely** positive maps.

We will discuss the special properties of completely positive trace preserving maps, describe the most common examples, and, last but not least, specify when the action of quantum channels can be reversed, or corrected, so that we can recover the original input state. This will set the stage for our subsequent discussion of quantum error correction.

9.2 Random unitaries

As a first step toward understanding the quantum description of an evolving open system, consider a “two-qubit universe” in which we observe *only one* of the qubits. Let’s revisit the controlled-NOT gate, in which two qubits undergo the unitary transformation

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

but we’re going to focus on the transformation of the target qubit alone. We know that it depends on the state of the control qubit:

- if the input state of the control qubit is $|0\rangle$, the target qubit evolves (*unitarily*) according to the identity operator $\mathbf{1}$;
- if the input state of the control qubit is $|1\rangle$, the target qubit evolves (*unitarily*) according to the bit-flip operator X ;
- ... but for input states of the control that are *superpositions* of $|0\rangle$ and $|1\rangle$ the evolution of the target qubit is *not* unitary.

To justify this last point, note that, if the control qubit is in the state $\alpha_0|0\rangle + \alpha_1|1\rangle$ and the target qubit is in some state $|\psi\rangle$, then the output state can be written as

$$\alpha_0|0\rangle \otimes \mathbf{1}|\psi\rangle + \alpha_1|1\rangle \otimes X|\psi\rangle$$

which shows that the control and the target become entangled. The target qubit alone ends up in the statistical mixture of states $|\psi\rangle$ with probability $|\alpha_0|^2$ and $X|\psi\rangle$ with probability $|\alpha_1|^2$.

We can verify this by expressing the above output state of the two qubits as the density matrix

$$\begin{aligned} |\alpha_0|^2|0\rangle\langle 0| \otimes \mathbf{1}|\psi\rangle\langle\psi|\mathbf{1} &+ |\alpha_1|^2|1\rangle\langle 1| \otimes X|\psi\rangle\langle\psi|X \\ + \alpha_0\alpha_1^*|0\rangle\langle 1| \otimes \mathbf{1}|\psi\rangle\langle\psi|X &+ \alpha_0^*\alpha_1|1\rangle\langle 0| \otimes X|\psi\rangle\langle\psi|\mathbf{1} \end{aligned}$$

and then tracing over the control qubit, which gives

$$|\alpha_0|^2 \mathbf{1} |\psi\rangle\langle\psi| \mathbf{1} + |\alpha_1|^2 X |\psi\rangle\langle\psi| X.$$

Then we can say that the input state of the target qubit evolves either according to the identity operator (with probability $|\alpha_0|^2$) or according to the X operator (with probability $|\alpha_1|^2$).

This argument works even if the target qubit is initially in a mixed state: we are dealing with a linear transformation, and any mixed state can be expressed as a statistical ensemble of pure states (via the convex decomposition $\rho = \sum_i p_i |\psi_i\rangle\langle\psi_i|$ of a density matrix). So, in general, we can express the evolution of the target qubit as

$$\rho \mapsto \rho' = |\alpha_0|^2 \mathbf{1} \rho \mathbf{1} + |\alpha_1|^2 X \rho X$$

where ρ and ρ' are the input and the output states, respectively. We may think about this input-output relation as a mathematical representation of a quantum communication channel in which an input qubit is bit-flipped (via the operator X) with some prescribed probability $|\alpha_1|^2$. But we may also take a more “global” view and see the action of the channel as arising from a unitary evolution on a larger (dilated) system, here composed of two qubits (namely the target *and* the control).

Our discussion can easily be extended beyond two qubits to cover any conditional dynamics of the type

$$U = \sum_i |i\rangle\langle i| \otimes U_i = \begin{bmatrix} U_1 & 0 & 0 & \dots \\ 0 & U_2 & 0 & \dots \\ 0 & 0 & U_3 & \dots \\ \vdots & \vdots & \vdots & \ddots \end{bmatrix}$$

where the vectors $|i\rangle$ form an orthonormal basis in the Hilbert space associated with a control system, and the U_i are the corresponding unitary operations performed on a target system. If the control system is prepared in state $\sum_i \alpha_i |i\rangle$ and the target in state $|\psi\rangle$, then the final state of the two systems is

$$\sum_i \alpha_i |i\rangle \otimes U_i |\psi\rangle$$

and, by the same sequence of arguments as before, we obtain the evolution of the target system alone, and express it as

$$\rho \mapsto \rho' = \sum_{i=1} |\alpha_i|^2 U_i \rho U_i^\dagger.$$

That is, the state of the target system is modified by the unitary U_i chosen randomly with probability $p_i = |\alpha_i|^2$.

The reason we are paying particular attention to random unitaries is that each unitary is invertible, and, as such, offers a sliver of hope for being able to *reverse* the overall action of the channel. Indeed, if we can learn, post factum, which particular unitary operation U_i was chosen, then we can simply apply the inverse $U_i^{-1} = U_i^\dagger$ of that unitary and recover the original state. For example, if we can measure the control system in the $|i\rangle$ basis, then measuring the outcome to be k tells us that we have to apply U_k^\dagger to the target to recover its input state.

However, if we do not have access to the control system, then there is very little we can do: *we cannot figure out which particular unitary was applied by inspecting the target system alone*. In this case the best we can do is to apply the inverse of the *most likely* unitary, which will then recover the input state, *but only with some probability of success*. In order to do better than that we have to look at slightly different channels.

First though, a fundamental example of a random unitary evolution:

Recall that, for the basis states, $\text{tr } |i\rangle\langle j| = \langle i|j\rangle = \delta_{ij}$.

We can also focus on the evolution of the control qubit: see Exercise 9.12.5. In fact, we can choose any subset of qubits for our inputs and outputs. For example, our input could be the control qubit, and the output could be *both* the control *and* the target qubits.

A **single-qubit Pauli channel** applies one of the Pauli operators, X , Y or Z , chosen randomly with some prescribed probabilities p_x , p_y and p_z , giving

$$\rho \longmapsto p_0 \mathbf{1} \rho \mathbf{1} + p_x X \rho X + p_y Y \rho Y + p_z Z \rho Z.$$

The Pauli operators represent **quantum errors**: bit-flip X , phase-flip Z , and the composition of the two $Y = iXZ$.

9.3 Random isometries

In many applications, including quantum communication and quantum error correction, it is useful to encode a quantum state of one system into a quantum state of a larger system. Such operations are described by *isometries*. You may think about isometries as a generalisation of unitaries: like unitaries, they preserve inner products; unlike unitaries, they are maps between spaces of *different* dimensions.

Let \mathcal{H} and \mathcal{H}' be Hilbert spaces such that $\dim \mathcal{H} \leq \dim \mathcal{H}'$. An **isometry** is a linear map $V: \mathcal{H} \rightarrow \mathcal{H}'$ such that $V^\dagger V = \mathbf{1}_{\mathcal{H}}$

Isometries preserve inner products, and therefore also the norm and the metric induced by the norm.

The word isometric (like pretty much most of the fancy words you come across in this course) comes from Greek, meaning “of the same measures”: *isos* means “equal”, and *metron* means “a measure”, and so an “isometry” is a transformation that preserves distances.

An isometry $V: \mathcal{H} \rightarrow \mathcal{H}'$ maps the *whole* Hilbert space \mathcal{H} onto a *subspace* of \mathcal{H}' . As a consequence, the matrix representation of an isometry is a rectangular matrix formed by selecting only a few of the columns from a unitary matrix. For example, given a unitary U we can construct an isometry V as follows:

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

The fact that an isometry V preserves the inner products comes from the fact that we require $V^\dagger V = \mathbf{1}_{\mathcal{H}}$; we do *not* require $VV^\dagger = \mathbf{1}_{\mathcal{H}'}$. Indeed, if we required both of these, then that would be equivalent to asking for V to be *unitary*. The operator VV^\dagger is a projector operator acting on \mathcal{H}' , which projects onto the image of \mathcal{H} under the isometry V , as we can see by expressing V in Dirac notation:

$$V = \sum_i |b_i\rangle\langle a_i|,$$

where the $|a_i\rangle$ form an orthonormal basis in \mathcal{H} , and the $|b_i\rangle$ are just orthonormal (but not necessarily spanning) vectors in \mathcal{H}' ; in the special case where V is unitary, the orthonormal vectors $|b_i\rangle$ form an orthonormal basis in \mathcal{H}' . Writing V in this form, it is clear that $V^\dagger V = \sum_i |a_i\rangle\langle a_i| = \mathbf{1}$, and that $VV^\dagger = \sum_i |b_i\rangle\langle b_i|$ projects on the subspace spanned by $|b_i\rangle$.

Although isometries are strictly more general than unitaries, an fundamentally important fact is that *isometries still represent physically admissible operations*: they can be implemented by bringing two systems together (via tensoring) and then applying unitary transformations to the composite system. That is, take some system \mathcal{A} in state $|\psi\rangle$, and bring in another system \mathcal{B} in some fixed state $|b\rangle$; applying some unitary U to the combined system \mathcal{AB} then gives an isometry from $\mathcal{H} = \mathcal{H}_{\mathcal{A}}$ to $\mathcal{H}' = \mathcal{H}_{\mathcal{A}} \otimes \mathcal{H}_{\mathcal{B}}$, i.e. the result is a linear map V defined by

$$V: |\psi\rangle \longmapsto |\psi\rangle|b\rangle \longmapsto U(|\psi\rangle|b\rangle).$$

Any isometry is a quantum channel, since any quantum state described by the state vector $|\psi\rangle$ (or by a density operator ρ) is transformed as

$$|\psi\rangle \mapsto V|\psi\rangle$$

(or as $\rho \mapsto V\rho V^\dagger$), and the normalisation condition is exactly the defining property of isometries:

$$V^\dagger V = \mathbf{1}.$$

Isometries are incredibly important when it comes to error correction, and we will see them again much more in Section ??.

9.4 Evolution of open systems

Needless to say, there is more to evolutions of open systems than mere random isometries, and what follows is the most general scenario that we will come across in our study of quantum information.

Consider two interacting systems, \mathcal{A} and \mathcal{B} , but this time do not assume that their interacting dynamics admits a control-target interpretation. We will view \mathcal{A} as an auxiliary system, i.e. an ancilla, and focus on the evolution of system \mathcal{B} .

Let us pick an orthonormal basis $|i\rangle$ of the Hilbert space $\mathcal{H}_\mathcal{A}$ associated with the ancilla. Any unitary transformation of the combined system \mathcal{AB} can then be written as

$$U = \sum_{i,j} |i\rangle\langle j| \otimes B_{ij} = \begin{bmatrix} B_{11} & B_{12} & B_{13} & \dots \\ B_{21} & B_{22} & B_{23} & \dots \\ B_{31} & B_{32} & B_{33} & \dots \\ \vdots & \vdots & \vdots & \ddots \end{bmatrix}$$

where the B_{ij} are operators acting on the the Hilbert space $\mathcal{H}_\mathcal{B}$ associated with system \mathcal{B} . Note that the B_{ij} do not need to be unitary, but, for the overall transformation U to be unitary, they must satisfy

$$\begin{aligned} \sum_i B_{ik}^\dagger B_{il} &= \delta_{kl} \mathbf{1}_{\mathcal{AB}} \\ \sum_i B_{ki} B_{li}^\dagger &= \delta_{kl} \mathbf{1}_\mathcal{B} \end{aligned} \tag{\star}$$

where $\mathbf{1}_{\mathcal{AB}}$ and $\mathbf{1}_\mathcal{B}$ are the identity operators on $\mathcal{H}_\mathcal{A} \otimes \mathcal{H}_\mathcal{B}$ and $\mathcal{H}_\mathcal{B}$, respectively. These two conditions correspond to the requirement that both column and row vectors must be orthonormal for a matrix to be unitary, except that here U is a block matrix, and the entries B_{ij} are complex matrices rather than complex numbers, so some care must be taken with the order of multiplication. Again, the evolution of the system \mathcal{B} depends on both U and on the initial state of the auxiliary system \mathcal{A} .

Without any loss of generality, we may assume that system \mathcal{A} is in a pure state, which can be chosen to be one of the basis states $|i\rangle$, say $|k\rangle$. In this case, U acts by

$$U: |k\rangle \otimes |\psi\rangle \mapsto \sum_i |i\rangle \otimes B_{ik} |\psi\rangle \tag{\ddagger}$$

for an arbitrary state $|\psi\rangle$ of \mathcal{B} .

The resulting density operator for \mathcal{B} is found by taking the density operator of the output state of \mathcal{AB} , which is

$$\sum_{i,j} |i\rangle\langle j| \otimes B_{ik} |\psi\rangle\langle\psi| B_{jk}^\dagger$$

and then tracing out \mathcal{A} , obtaining

For now, when we write tensor products, we will place the ancilla first and the system of interest second: $\mathcal{H}_\mathcal{A} \otimes \mathcal{H}_\mathcal{B}$. We do this to begin with simply because block matrices on tensor products are easier to interpret when written in this particular order. Later on we will revert to the more common convention in which the system of interest is placed first.

If \mathcal{A} were initially in a mixed state, we could always regard \mathcal{A} as a subsystem of some larger $\tilde{\mathcal{A}}$ that is in an entangled pure state.

Recall that $\langle i|j\rangle = \delta_{ij}$.

$$\begin{aligned}\mathrm{tr}_{\mathcal{A}} \left(\sum_{i,j} |i\rangle\langle j| \otimes B_{ik} |\psi\rangle\langle\psi| B_{jk}^\dagger \right) &= \sum_{i,j} \langle i|j\rangle \cdot B_{ik} |\psi\rangle\langle\psi| B_{jk}^\dagger \\ &= \sum_i B_{ik} |\psi\rangle\langle\psi| B_{ik}^\dagger.\end{aligned}$$

In general, for any input state ρ , we obtain the map

$$\begin{aligned}\rho &\longmapsto \rho' = \sum_i B_{ik} \rho B_{ik}^\dagger \\ &=: \sum_i B_i \rho B_i^\dagger\end{aligned}$$

where, in the last expression on the right-hand size, we have dropped index k (remember, it was there only to remind us about the initial state of the ancilla). Since the overall transformation U is unitary, recall that the B_i satisfy $\sum_i B_i^\dagger B_i = \mathbf{1}$. This normalisation conditions guarantees that the trace is preserved.

In summary, we can think about a quantum evolution of subsystem \mathcal{B} as a sequence of the three distinct operations:

$$\begin{aligned}\rho &\longmapsto \underbrace{|k\rangle\langle k| \otimes \rho}_{\text{add ancilla}} \\ &\longmapsto \underbrace{U(|k\rangle\langle k| \otimes \rho)U^\dagger}_{\text{unitary evolution}} \\ &\longmapsto \underbrace{\mathrm{tr}_{\mathcal{A}} [U(|k\rangle\langle k| \otimes \rho)U^\dagger]}_{\text{discard ancilla}} = \sum_i B_i \rho B_i^\dagger = \rho'.\end{aligned}$$

In words:

- First we pick up a system of interest which, in general, can be in a mixed state ρ . It may be the case that this system is entangled with some other degrees of freedom or with some other physical systems, but these other entities will remain passive and will not enter any subsequent dynamics.
- Then we dilate the system: we add an ancilla which is large enough to include everything our system will interact with, and also large enough to be in a pure state. The expansion ends when the composed system is (for all practical purposes) isolated and follows a unitary evolution U .
- We allow the expanded system to evolve under the unitary evolution.
- After the unitary evolution takes place, we discard the ancilla and focus on the system alone. In fact we do not have to discard exactly what we added: we can discard only part of the ancilla, or any other part of the dilated system.

It is adding (i.e. tensoring) the auxiliary system in a fixed state, and then discarding it (via the partial trace), that is responsible for the seemingly *non-unitary* character of this evolution.

The next step is to use what we have learnt about isometries (namely that they are like unitaries but where the dimension is allowed to increase) to combine the first two of these operations (adding an ancilla and following some unitary evolution) into a single operation. This will lead to the so-called **Stinespring dilation theorem**, as well as its ancilla-free counterpart, the **Kraus decomposition**.

Because of this, the output system in this scenario does not have to be the same as the original input system (e.g. it could be strictly larger), but usually it is.

Factorisation systems.

This three-stage process (adding an ancilla, unitary evolution, and then tracing out the ancilla) might reasonably be called a “factorisation”, since it factors a (non-unitary) evolution into constituent parts: first something that looks a bit like an injection (since it maps a smaller space into a bigger one); then something that looks a bit like an isomorphism (since unitaries are invertible); and finally something that looks a bit like a surjection (since it maps a bigger space down to a smaller one). For now, let’s forget about this middle part of the factorisation (where we let our system evolve unitarily), and just keep the first and last part in mind as we look at the following construction.

Pick any function $f: S \rightarrow T$ between sets. Then we can decompose f into an injection (\hookrightarrow) and a surjection (\twoheadrightarrow) in two different ways:

1. $S \twoheadrightarrow \text{Im}(f) \hookrightarrow T$
2. $S \hookrightarrow S \sqcup (T \setminus \text{Im}(f)) \twoheadrightarrow T$

where the first is a surjection followed by an injection, and the second is an injection followed by a surjection. In the first decomposition, the middle set (namely $\text{Im}(f)$) is unique (up to unique isomorphism); in the second, the middle set (namely $S \sqcup (T \setminus \text{Im}(f))$) is *not* unique (we can use any set given by taking S and adding an extra arbitrary element for each element of T that is not in the image of f).

The first of these decompositions is probably much more familiar and friendly looking than the second, but it is indeed the second which is of interest to us here, since it is of the same form as our three-stage process: something injective-looking followed by something surjective-looking. Indeed, as shown in Cunningham and Heunen’s “Purity through Factorisation”, arXiv:[1705.07652](https://arxiv.org/abs/1705.07652), Stinespring dilation (which is roughly this three-stage process that we’ve been talking about) gives rise to a **weak factorisation system**, but *not* an **orthogonal** one.

These notions (weak and orthogonal factorisation systems) are absolutely fundamental to a large area of modern mathematics that deals with homotopy theory and “higher structures” using the language of [model categories](#).

9.5 Stinespring's dilation and Kraus's ambiguity

Once we start playing with adding physical systems and increasing the dimension of the underlying Hilbert space, it is convenient to switch from unitaries to isometries. This is more for mathematical simplicity than physical insight, but it is always good to declutter our equations a bit if we can.

Recall that any unitary transformation of the combined system $\mathcal{A}\mathcal{B}$ can be written as

$$U = \sum_{i,j} |i\rangle\langle j| \otimes B_{ij} = \begin{bmatrix} B_{11} & B_{12} & B_{13} & \dots \\ B_{21} & B_{22} & B_{23} & \dots \\ B_{31} & B_{32} & B_{33} & \dots \\ \vdots & \vdots & \vdots & \ddots \end{bmatrix}$$

where the B_{ij} are operators acting on the the Hilbert space $\mathcal{H}_{\mathcal{B}}$, and where the B_{ij} are *not* necessarily unitary, but (in order for the overall transformation U to be unitary) satisfy

$$\sum_i B_{ik}^\dagger B_{il} = \delta_{kl} \mathbf{1}_{\mathcal{A}\mathcal{B}}$$

$$\sum_i B_{ki} B_{li}^\dagger = \delta_{kl} \mathbf{1}_{\mathcal{B}}$$

Recall that a map V is an isometry if $V^\dagger V = \mathbf{1}$. For example, adding a system in state $|k\rangle$ gives an isometry $V: |\psi\rangle \mapsto |k\rangle \otimes |\psi\rangle$, and the combination of adding a system in a fixed state followed by a unitary evolution of the combined system is also an isometry.

Also recall that, when we fix the initial state of system \mathcal{A} to be $|k\rangle$, we know that U acts by

$$U: |k\rangle \otimes |\psi\rangle \mapsto \sum_i |i\rangle \otimes B_{ik}|\psi\rangle$$

for an arbitrary state $|\psi\rangle$ of \mathcal{B} .

This allows us to define an isometry $V: \mathcal{H}_{\mathcal{B}} \rightarrow \mathcal{H}_{\mathcal{A}} \otimes \mathcal{H}_{\mathcal{B}}$ by

$$V: |\psi\rangle \mapsto \sum_i |i\rangle \otimes E_i|\psi\rangle$$

where $E_i := B_{ik}$, which satisfy

$$\sum_i E_i^\dagger E_i = \mathbf{1}_{\mathcal{B}}.$$

The matrix representation of an isometry is a rectangular matrix given by selecting only a few of the columns from a unitary matrix; here, with $|k\rangle$ fixed, it is only the k -th column of the block matrix U that determines the evolution of \mathcal{B} , as shown in Figure 9.1.

$$U = \begin{bmatrix} B_{11} & B_{12} & B_{13} & \dots \\ B_{21} & B_{22} & B_{23} & \dots \\ B_{31} & B_{32} & B_{33} & \dots \\ \vdots & \vdots & \vdots & \ddots \end{bmatrix} \mapsto V = \begin{bmatrix} E_1 \\ E_2 \\ E_3 \\ \vdots \end{bmatrix}$$

Figure 9.1: For $k = 2$, the second block column is selected. The matrix representation of the isometry V on the right-hand side look like a column vector, but remember that the entries $E_i := B_{ik}$ are *matrices*.

Let us now rephrase our derivation of the evolution of system \mathcal{B} using isometries. Note that the isometry V in Figure 9.1 acts by

$$|\psi\rangle\langle\psi| \mapsto V|\psi\rangle\langle\psi|V^\dagger = \sum_{i,j} |i\rangle\langle j| \otimes E_i|\psi\rangle\langle\psi|E_j^\dagger.$$

We trace out \mathcal{A} (recalling that $\text{tr}|i\rangle\langle j| = \langle i|j\rangle = \delta_{ij}$) and express the evolution of system \mathcal{B} (which is allowed to have a mixed input state ρ , since these can always be expressed as statistical mixtures of pure states $|\psi\rangle$) as

$$\rho \mapsto \rho' = \text{tr}_{\mathcal{A}} V \rho V^\dagger = \sum_i E_i \rho E_i^\dagger,$$

where $\sum_i E_i^\dagger E_i = \mathbf{1}$. This expression shows two different ways of looking at quantum evolutions, and both have their own name.

Stinespring dilation. Any quantum channel \mathcal{E} can be thought of as arising from a *unitary* evolution on a *dilated* system. When we combine tensoring and the unitary evolution into an isometry V , we can express the action of the channel \mathcal{E} as

$$\rho \mapsto \rho' = \text{tr}_{\mathcal{A}} V \rho V^\dagger,$$

where we trace out a suitably chosen ancilla \mathcal{A} . This is the approach that we discussed in Section 9.4. In quantum information science, we often refer to this approach as the **Church of the Larger Hilbert Space**.

William Forrest "Woody" Stinespring (1929–2012) was an American mathematician specialising in operator theory. Karl Kraus (1938–1988) was a German physicist known for his contributions to the mathematical foundations of quantum theory. His book *States, effects, and operations* (Lecture Notes in Physics, Vol. 190, Springer-Verlag, Berlin 1983) is an early account of the notion of complete positivity in physics.

Kraus representation (a.k.a. **operator-sum decomposition**). It is often more convenient to not deal with a larger Hilbert space, but to instead work with operators directly between the input and output Hilbert spaces, avoiding the middle one completely:

$$\rho \longmapsto \rho' = \sum_i E_i \rho E_i^\dagger$$

where the **Kraus operators** (or **effects**) E_i satisfy the normalisation condition $\sum_i E_i^\dagger E_i = \mathbf{1}$ (also known as the **completeness relation**). Here we avoid dragging in the ancilla, which can be a good thing, since ancillas typically represent environments that can be very large and complex. Note that this operator-sum decomposition is *not unique*, since the Kraus operators E_i depend on the choice of basis in the ancilla.

These two representations are equivalent, and we can easily switch between them:

- We have already seen how to go from a unitary evolution U on a larger system to an isometry V , and then to a map on density operators represented by a set of Kraus operators E_i (as in Figure 9.1).
- Conversely, once we have an operator-sum representation of the channel with a set of Kraus operators E_i , we can introduce an ancilla of dimension equal to the number of Kraus operators, and use the orthonormal basis $|i\rangle$ to form the isometry $V = \sum_i |i\rangle \otimes E_i$. In terms of matrices, this corresponds to simply “stacking up” the matrices E_i to form the block column (as shown in Figure 9.1), which gives us the matrix representation of V . If we want to go further, from an isometry V to a unitary U , then the next step is somewhat arbitrary: we can choose all the remaining block columns of U however we please, *as long as* we end up with a unitary matrix U .

All linear transformations of density operators that can be written in Stinespring (or, equivalently, Kraus) form represent *physically realisable operations* — we call them **quantum channels**, or **superoperators** (since they send operators to operators).

We note again that the Kraus decomposition is *not unique*: the operators E_i depend on the choice of the ancilla basis. Indeed, let $|e_i\rangle$ and $|f_j\rangle$ be two orthonormal bases in the Hilbert space associated with the ancilla. Then V can be expressed as

$$\begin{aligned} V &= \sum_i |e_i\rangle \otimes E_i \\ &= \sum_{i,j} |f_j\rangle \langle f_j| e_i \rangle \otimes E_i \\ &= \sum_j |f_j\rangle \otimes \sum_i \underbrace{\langle f_j| e_i \rangle}_{R_{ji}} E_i \\ &= \sum_j |f_j\rangle \otimes F_j \end{aligned}$$

where we have used the fact that $\sum_j |f_j\rangle \langle f_j| = \mathbf{1}$, and where $R_{ji} = \langle f_j| e_i \rangle$ is a unitary matrix connecting the two orthonormal bases (and also the two sets of the Kraus operators) via $F_j = \sum_i R_{ji} E_i$. So we have a set of Kraus operators E_i associated with basis $|e_i\rangle$ and another, unitarily related, set of Kraus operators F_j associated with basis $|f_j\rangle$, and the two sets describe the same isometry, and hence the same quantum

channel. This correspondence goes both ways: if two channels \mathcal{E} and \mathcal{F} have their Kraus operators related by some unitary R_{ji} , then the two channels are identical:

$$\begin{aligned}\mathcal{F}(\rho) &= \sum_j F_j \rho F_j^\dagger \\ &= \sum_{i,j,k} R_{ji} E_i \rho E_k^\dagger R_{jk}^* \\ &= \sum_{i,k} \underbrace{\left(\sum_j R_{jk}^* R_{ji} \right)}_{\delta_{ki}} E_i \rho E_k^\dagger \\ &= \sum_i E_i \rho E_i^\dagger \\ &= \mathcal{E}(\rho).\end{aligned}$$

In summary:

Suppose E_1, \dots, E_n and F_1, \dots, F_m are Kraus operators associated with quantum channels \mathcal{E} and \mathcal{F} , respectively. We can append zero operators to the shorter list to ensure that $n = m$ (or we could view R_{ij} as an *isometry* instead of a unitary).

Then \mathcal{E} and \mathcal{F} describe the same channel if and only if $F_j = \sum_i R_{ji} E_i$ for some unitary R .

In particular, this unitary equivalence of the Kraus operators implies that the identity channel $\rho \mapsto \rho' = \mathbf{1}\rho\mathbf{1}$ can only have Kraus operators that are proportional to the identity.

9.6 Single-qubit channels

The best way to familiarise ourselves with the concept of a quantum channel is to study a few examples, and we will start with the simplest case: **single-qubit channels**. The single-qubit case is special since we can visualise the action of the channel by looking at the corresponding deformation of the Bloch ball.

Recall that an arbitrary density matrix for a single qubit can be written in the form

$$\begin{aligned}\rho &= \frac{1}{2} (\mathbf{1} + \vec{s} \cdot \vec{\sigma}) \\ &= \frac{1}{2} (\mathbf{1} + s_x X + s_y Y + s_z Z)\end{aligned}$$

where \vec{s} is the Bloch vector of the qubit with components (s_x, s_y, s_z) , and X , Y , and Z are the Pauli operators. Recall also that unitary operations *rotate* the Bloch sphere. In particular the X , Y , and Z operators — viewed as unitary transformations — rotate the Bloch sphere by 180° around the x -, y -, and z -axis, respectively. General quantum channels, however, may deform it further, into spheroids with a displaced centre, as the following examples show.

- Bit-flip with probability p .

$$\rho \mapsto (1-p)\rho + pX\rho X.$$

The Kraus operators are $\sqrt{1-p}\mathbf{1}$ and $\sqrt{p}X$; the original Bloch sphere shrinks into a prolate spheroid aligned with the x -axis; for the specific case of $p = \frac{1}{2}$, the Bloch sphere degenerates to the $[-1, 1]$ interval on the x -axis.

- **Phase-flip with probability p .**

$$\rho \mapsto (1-p)\rho + pZ\rho Z.$$

The Kraus operators are $\sqrt{1-p}\mathbf{1}$ and $\sqrt{p}Z$; the original Bloch sphere shrinks into a prolate spheroid aligned with the z -axis; for the specific case of $p = \frac{1}{2}$, the Bloch sphere degenerates to the $[-1, 1]$ interval on the z -axis.

- **Depolarising channel with probability p .**

$$\rho \mapsto (1-p)\rho + \frac{p}{3}(X\rho X + Y\rho Y + Z\rho Z).$$

Here the qubit remains intact with probability $1 - p$, while a quantum error occurs with probability p . The error can be of any one of three types: bit-flip X , phase-flip Z , or both bit- and phase-flip Y ; each type of error is equally likely. For $p < \frac{3}{4}$, the original Bloch sphere contracts uniformly under the action of the channel, and the Bloch vector shrinks by the factor $1 - \frac{4}{3}p$; for the specific case of $p = \frac{3}{4}$, the Bloch sphere degenerates to the point at the centre of the sphere; for $p > \frac{3}{4}$, the Bloch sphere is flipped, and the Bloch vector starts pointing in the opposite direction increasing the magnitude up to $\frac{1}{3}$ (which occurs for $p = 1$).

There are two interesting points that must be mentioned here. The first one is about the interpretation of the action of the channel in terms of Kraus operators: our narrative may change when we switch to a different set of effects. For example, take the phase-flip channel with $p = \frac{1}{2}$ and switch from the effects E_i to F_j as follows:

$$\left\{ \begin{array}{l} E_1 = \frac{1}{\sqrt{2}}\mathbf{1} \\ E_2 = \frac{1}{\sqrt{2}}Z \end{array} \right\} \mapsto \left\{ \begin{array}{l} F_1 = \frac{1}{\sqrt{2}}(E_1 + E_2) = |0\rangle\langle 0| \\ F_2 = \frac{1}{\sqrt{2}}(E_1 - E_2) = |1\rangle\langle 1|. \end{array} \right\}$$

These two sets of Kraus operators $\{E_1, E_2\}$ and $\{F_1, F_2\}$ describe the same channel, but the *narrative* is different: the first set of effects tells us that the channel chooses randomly, with the same probability, between two options (let the qubit pass undisturbed or apply the phase-flip Z); the second set tells us that channel essentially performs the measurement in the standard basis, but the outcome of the measurement is not revealed.

Recall that Kraus operators are also sometimes called “effects”.

Describing actions of quantum channels purely in terms of their effects (i.e. Kraus operators) can be ambiguous.

The second interesting point is that not *all* transformations of the Bloch sphere into spheroids are possible. For example, we cannot deform the Bloch sphere into a pancake-like oblate spheroid. This is due to *complete positivity* (instead of mere positivity) of quantum channels, which we will explain shortly.

9.7 Composition of quantum channels

We mentioned that quantum channels are combinations of

1. adding a physical system in a fixed state (via tensoring),
2. unitary transformations, and
3. discarding a physical system (taking a partial trace).

As expected from the fact that the Stinespring point of view is equivalent to the Kraus point of view, each of these operations admits an operator-sum decomposition. This is obvious for unitary evolution ($\rho \mapsto U\rho U^\dagger$), but perhaps less so for the other two operations. One reason to care about this is that the Kraus decomposition gives a tidy way of describing *composition* of quantum channels.

- **Adding a system.** Any quantum system can be expanded by bringing in an auxiliary system in a fixed state $|a\rangle$. This transformation takes vectors in the Hilbert space associated with the original system and tensors them with a fixed vector $|a\rangle$ in the Hilbert space associated with the auxiliary system:

$$|\psi\rangle \mapsto |a\rangle \otimes |\psi\rangle = (|a\rangle \otimes \mathbf{1})|\psi\rangle.$$

In terms of density operators, we write this “expansion” transformation as

$$\begin{aligned} \rho &\mapsto \rho' = |a\rangle\langle a| \otimes \rho \\ &= (|a\rangle \otimes \mathbf{1})\rho(\langle a| \otimes \mathbf{1}) \\ &= V\rho V^\dagger \end{aligned}$$

where $V = |a\rangle \otimes \mathbf{1}$. We note that $V^\dagger V = \langle a|a\rangle \otimes \mathbf{1} = \mathbf{1}$ is the identity in the Hilbert space associated with the system, and so V is an isometry. Indeed, this transformation is an *isometric embedding*.

- **Discarding a system.** Conversely, given a composite system in state ρ , we can discard one of its subsystems. The partial trace over an auxiliary system can be written in the Kraus representation as

$$\begin{aligned} \rho &\mapsto \rho' = \text{tr}_{\mathcal{A}} \rho \\ &= (\text{tr} \otimes \mathbf{1})\rho \\ &= \sum_i (\langle i| \otimes \mathbf{1})\rho(|i\rangle \otimes \mathbf{1}) \\ &= \sum_i E_i \rho E_i^\dagger \end{aligned}$$

where the vectors $|i\rangle$ form an orthonormal basis in the Hilbert space associated with the auxiliary system. Again, we can check that the Kraus operators $E_i = |i\rangle \otimes \mathbf{1}$ satisfy the completeness relation $\sum_i E_i^\dagger E_i = \mathbf{1} \otimes \mathbf{1}$ (using the fact that $\sum_i |i\rangle\langle i| = \mathbf{1}$).

Any **sequential** composition of two quantum channels \mathcal{E} and \mathcal{F} with Kraus operators $\{A_i\}_{i \in I}$ and $\{B_j\}_{j \in J}$ (respectively) is another quantum channel described by the Kraus operators $\{B_j A_i\}_{i \in I, j \in J}$. Showing this is rather straightforward, at least in the operator-sum representation: let

$$\begin{aligned} \mathcal{E} &= \sum_i A_i \cdot A_i^\dagger \\ \mathcal{F} &= \sum_j B_j \cdot B_j^\dagger \end{aligned}$$

where $\sum_i A_i^\dagger A_i = \sum_j B_j^\dagger B_j = \mathbf{1}$; then the sequential composition of \mathcal{E} followed by \mathcal{F} can be written as

$$\mathcal{F} \circ \mathcal{E} = \sum_{i,j} (B_j A_i) \cdot (B_j A_i)^\dagger$$

so that the $B_j A_i$ are the Kraus operators associated with the new channel $\mathcal{F} \circ \mathcal{E}$, where the normalisation condition (or completeness relation) follows from

$$\begin{aligned} \sum_{i,j} (B_j A_i)^\dagger (B_j A_i) &= \sum_i A_i^\dagger \left(\sum_j B_j^\dagger B_j \right) A_i \\ &= \sum_i A_i^\dagger A_i \\ &= \mathbf{1}. \end{aligned}$$

Here we have tacitly assumed that the dimensions agree, i.e. that the output of \mathcal{E} and the input of \mathcal{F} are of the same dimension, so that the composition makes sense.

You might wonder why we explicitly called the above composition “sequential”—isn’t this how we always compose functions? In actual fact, since we have access to tensor products, there is another sort of composition, namely **parallel** composition: if we have systems \mathcal{A} and \mathcal{B} with channels $\mathcal{E}_{\mathcal{A}}$ acting on \mathcal{A} and $\mathcal{E}_{\mathcal{B}}$ acting on \mathcal{B} , then the parallel composition is denoted by $\mathcal{E}_{\mathcal{A}} \otimes \mathcal{E}_{\mathcal{B}}$, acting on the joint system $\mathcal{A} \otimes \mathcal{B}$, and with Kraus operators given by the $A_i \otimes B_j$. The normalisation condition again follows from a simple calculation:

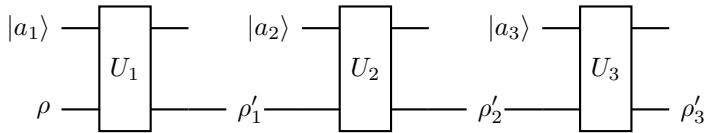
$$\begin{aligned}\sum_{i,j} (A_i \otimes B_j)^\dagger (A_i \otimes B_j) &= \sum_{i,j} A_i^\dagger A_i \otimes B_j^\dagger B_j \\ &= \mathbf{1}_A \otimes \mathbf{1}_B.\end{aligned}$$

Now that we know how to compose quantum channels in terms of Kraus operators, we can see that the Stinespring representation is perfectly consistent with the Kraus representation: the three basic operations that we are allowed to use to build channels in the Stinespring representation (i.e. adding a system, unitary evolution, and discarding a system) are all themselves quantum channels, in that they admit a Kraus decomposition.

Before moving on, we make a small (but important) remark:

When we compose quantum channels, each channel needs its own independent ancilla — *do not share ancillas between different channels*.

For example, say we have three channels, \mathcal{E}_1 , \mathcal{E}_2 , and \mathcal{E}_3 , with \mathcal{E}_i defined by the unitary U_i and the state $|a_i\rangle$ of its ancilla. Then the (sequential) composition $\mathcal{E}_3 \circ \mathcal{E}_2 \circ \mathcal{E}_1$ is given by



where each \mathcal{E}_i has its own associated ancilla $|a_i\rangle$. For more on this, see Section 9.12.2, where we talk about **Markov approximation**.

9.8 Completely positive trace-preserving maps

A while back we upgraded from working with state vectors $|\psi\rangle$ to working with density operators ρ , which are positive Hermitian operators ρ with $\text{tr } \rho = 1$, where “positive” means that $\langle v | \rho | v \rangle \geq 0$ for all $|v\rangle$ (or, equivalently, that all its eigenvalues are non-negative real numbers). It is easy to verify that quantum channels preserve positivity and trace, but the converse is not true! That is, there are linear maps that preserve positivity and the trace, but which are *not* quantum channels, and thus which are not “physical operations”.

The matrix transpose operation $\rho \mapsto \rho^T$ is a good example of such an unphysical operation: it preserves both trace and positivity, and if ρ is a density matrix then so too is ρ^T , but we will show that the transpose *cannot* be written in the Stinespring (or the Kraus) form; it is not induced by a unitary operation on some larger Hilbert space, and it cannot be physically implemented. So, we then ask, *what is* the class of physically admissible maps? That is, how can we classify which maps are quantum channels and which are not?

First, some notation. We say that a linear operator $f: \mathcal{H} \rightarrow \mathcal{H}'$ between Hilbert spaces is **bounded** if there exists some real number $B > 0$ such that $\|f(x)\|_{\mathcal{H}'} \leq B \|x\|_{\mathcal{H}}$ for every vector $x \in \mathcal{H}$. Given a pair of Hilbert spaces \mathcal{H} and \mathcal{H}' , we denote

You might also call this **simultaneous** composition, to contrast with *sequential* composition, but “parallel” is by far the most commonly accepted terminology.

It’s a small abuse of notation, but we often simply say “positive” to mean “positive semi-definite” or “non-negative”. We write $\rho \geq 0$ to mean that ρ is positive.

the set of bounded linear operators from \mathcal{H} to \mathcal{H}' by $\mathcal{B}(\mathcal{H}, \mathcal{H}')$. We write $\mathcal{B}(\mathcal{H})$ as shorthand for $\mathcal{B}(\mathcal{H}, \mathcal{H})$.

Bounded and unbounded operators.

One reason to care so much about *bounded* operators is the following fact: *a linear operator between normed vector spaces is bounded if and only if it is continuous.*

Another important fact is that the set $\mathcal{B}(\mathcal{H}, \mathcal{H}')$ is more than just a mere set: it has both topological structure (it has a norm and forms a Banach space under this norm) and algebraic structure (it is an associative algebra over \mathbb{C}), along with the bonus feature of a particularly well-behaved **involution** given by the **adjoint**. Formally, $\mathcal{B}(\mathcal{H}, \mathcal{H}')$ is the prototypical example of a **C^* -algebra**.

Now here is another example of where working only with finite-dimensional spaces greatly simplifies the mathematics: if X and Y are normed vector spaces, with X finite dimensional, then *every linear map $f: X \rightarrow Y$ is bounded (or, equivalently, continuous)*.

In the infinite-dimensional setting, it is important to know whether or not a given operator is bounded, but it turns out that certain unbounded operators are still very useful. There are some **technical details**, but such operators **are used to model observables** in the Hilbert-space formalism of quantum mechanics.

Then, mathematically speaking, a quantum channel \mathcal{E} is a specific type of map

$$\mathcal{E}: \mathcal{B}(\mathcal{H}) \rightarrow \mathcal{B}(\mathcal{H}')$$

that sends states (i.e. density operators) on some Hilbert space \mathcal{H} to states on some (possibly different) Hilbert space \mathcal{H}' . But we are not interested in just *any* such maps, of course — the statistical interpretation of quantum theory imposes certain properties on the subset of maps in which we are interested.

Firstly, for such a map \mathcal{E} to be a channel it must *respect the mixing of states*. Consider an ensemble of systems, with a fraction p_1 of them in the state ρ_1 , and the remaining p_2 of them in the state ρ_2 . The overall ensemble is described by $\rho = p_1\rho_1 + p_2\rho_2$. If we apply \mathcal{E} to each member of the ensemble individually, then the overall ensemble will be described by the density operator $\rho' = \mathcal{E}(\rho)$, which should be given by $\rho' = p_1\mathcal{E}(\rho_1) + p_2\mathcal{E}(\rho_2)$. We conclude that \mathcal{E} must be a *linear map*.

Next, since \mathcal{E} must *map density operators to density operators*, it has to be both *positive* ($\mathcal{E}(\rho) \geq 0$ whenever $\rho \geq 0$) and *trace preserving* ($\text{tr } \mathcal{E}(\rho) = \text{tr } \rho$ for all ρ).

Finally comes a subtle point. It turns out that being positive is not good enough; we must further require that the map \mathcal{E} *remains positive even when extended to act on a part of a larger system*. Suppose that Alice and Bob share a bipartite system $\mathcal{A}\mathcal{B}$ in an entangled state ρ_{AB} , and, whilst Alice does nothing, Bob applies the operation \mathcal{E} to his subsystems, and his subsystems only. Then the resulting map on the whole bipartite system is given by $1 \otimes \mathcal{E}$, and we require that this *also* give a density operator ρ'_{AB} of the composed system. It turns out that this is a *strictly stronger* property than mere positivity; we are asking for something called **complete positivity**. Needless to say, complete positivity of \mathcal{E} implies positivity, but the converse does not hold: there are maps which are positive but not completely positive. The matrix transpose operation $\rho \rightarrow \rho^T$ is a classic example of such a map.

Let's study this matrix transpose example a bit more. Consider the transpose operation on a single qubit: $T: |i\rangle\langle j| \mapsto |j\rangle\langle i|$ (for $i, j \in \{0, 1\}$). It preserves both trace and positivity, and if ρ is a density matrix then so too is $T(\rho) = \rho^T$. However, if the input qubit is part of a two qubit system, initially in the entangled state $|\Omega\rangle = \frac{1}{\sqrt{2}}(|0\rangle|0\rangle + |1\rangle|1\rangle)$, and the transpose is applied to *only one* of the two qubits

(say, the second one), then the density matrix of the two qubits evolves under the action of the **partial transpose** $\mathbf{1} \otimes T$ as

$$\begin{aligned} |\Omega\rangle\langle\Omega| &= \frac{1}{2} \sum_{i,j} |i\rangle\langle j| \otimes |i\rangle\langle j| \xrightarrow{\mathbf{1} \otimes T} \frac{1}{2} \sum_{i,j} |i\rangle\langle j| \otimes T(|i\rangle\langle j|) \\ &= \frac{1}{2} \sum_{i,j} |i\rangle\langle j| \otimes |j\rangle\langle i|. \end{aligned}$$

The output is known as the **SWAP** matrix, since it describes the SWAP operation: $|j\rangle|i\rangle \mapsto |i\rangle|j\rangle$. Since this operation squares to the identity, we know that its eigenvalues must be either ± 1 : states which are symmetric under interchange of the two qubits have eigenvalue 1, while antisymmetric states have eigenvalue -1 . In particular then, the SWAP matrix has negative eigenvalues, which means that $\mathbf{1} \otimes T$ does *not* preserve positivity (since $\mathbf{1} \otimes T$ applied to the positive operator $|\Omega\rangle\langle\Omega|$ is *not* positive), and therefore T is *not* a completely positive map.

If you prefer to see this more explicitly, then you can use the matrix representation of $|\Omega\rangle\langle\Omega|$, apply the partial transpose $\mathbf{1} \otimes T$, and then inspect the resulting matrix:

$$\frac{1}{2} \left[\begin{array}{cc|cc} 1 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 \\ \hline 0 & 0 & 0 & 0 \\ 1 & 0 & 0 & 1 \end{array} \right] \xrightarrow{\mathbf{1} \otimes T} \frac{1}{2} \left[\begin{array}{cc|cc} 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ \hline 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 \end{array} \right].$$

So the partial transpose $\mathbf{1} \otimes T$ maps the density matrix $|\Omega\rangle\langle\Omega|$ of a maximally mixed state $|\Omega\rangle$ to the SWAP matrix, which has a negative eigenvalue (namely -1) and thus is *not* a density matrix (since it is not positive).

We have seen that, at the very least, we want to be considering *completely positive trace-preserving* maps, but how do we know whether or not there are any further restrictions left to impose? Needless to say, here is where mathematics alone cannot guide us, since we are trying to characterise maps which are *physically admissible*, and mathematics knows nothing about the reality of our universe! However, one thing that we can do is compare our abstract approach with the derivations of quantum channels defined in terms of the Stinespring (or Kraus) representation. As it happens, we can (and will!) show that a map is completely positive and trace preserving *if and only if* it can be written in the Stinespring (or Kraus) form. In other words:

Quantum channels are *exactly* the completely positive trace-preserving (CPTP) maps.

One direction of this claim is much simpler than the other. Any quantum channel \mathcal{E} must be completely positive, since the Kraus decomposition guarantees positivity of both \mathcal{E} *and* the extended map $\mathbf{1} \otimes \mathcal{E}$, since if \mathcal{E} has Kraus decomposition $\sum_i E_i E_i^\dagger$, then the extended channel $\mathbf{1} \otimes \mathcal{E}$ has Kraus decomposition $\sum_i (\mathbf{1} \otimes E_i)(\mathbf{1} \otimes E_i^\dagger)$, which means that $\mathbf{1} \otimes \mathcal{E}$ is also a positive map, whence \mathcal{E} is completely positive.

Conversely, showing that CPTP maps are quantum channels is less simple. In order to prove this, we will now introduce a very convenient tool called the **Choi matrix**, which gives yet another way to characterise linear maps between operators.

9.9 Channel-state duality

Suppose that $\dim \mathcal{H} = d$ and $\dim \mathcal{H}' = d'$, and pick a basis for each space. Now any linear map $\mathcal{E}: \mathcal{B}(\mathcal{H}) \rightarrow \mathcal{B}(\mathcal{H}')$ can be completely characterised by its action on the d^2 -many basis matrices $|i\rangle\langle j|$ of $\mathcal{B}(\mathcal{H})$ (where $i, j \in \{1, 2, \dots, d\}$), i.e. for any density

operator ρ on \mathcal{H} we have

$$\mathcal{E}(\rho) = \mathcal{E}\left(\sum_{i,j=1}^d \rho_{ij}|i\rangle\langle j|\right) = \sum_{i,j=1}^d \rho_{ij}\mathcal{E}(|i\rangle\langle j|). \quad (\natural)$$

We can now tabulate the $(d \times d)$ -many $(d' \times d')$ matrices $\mathcal{E}(|i\rangle\langle j|)$ in \mathcal{H}' by forming a bigger $(dd' \times dd')$ block matrix in $\mathcal{H} \otimes \mathcal{H}'$:

$$\begin{bmatrix} \mathcal{E}(|0\rangle\langle 0|) & \mathcal{E}(|0\rangle\langle 1|) & \mathcal{E}(|0\rangle\langle 2|) & \cdots \\ \mathcal{E}(|1\rangle\langle 0|) & \mathcal{E}(|1\rangle\langle 1|) & \mathcal{E}(|1\rangle\langle 0|) & \cdots \\ \mathcal{E}(|2\rangle\langle 0|) & \mathcal{E}(|2\rangle\langle 1|) & \mathcal{E}(|2\rangle\langle 2|) & \cdots \\ \vdots & \vdots & \vdots & \ddots \end{bmatrix}.$$

After scaling by a factor of $\frac{1}{d}$, we call this block matrix $\tilde{\mathcal{E}} \in \mathcal{B}(\mathcal{H} \otimes \mathcal{H}')$ the **Choi matrix** of \mathcal{E} .

The Choi matrix is essentially another way of representing a linear map $\mathcal{E}: \mathcal{B}(\mathcal{H}) \rightarrow \mathcal{B}(\mathcal{H}')$, since if you are given the Choi matrix $\tilde{\mathcal{E}}$ of \mathcal{E} and you want to evaluate $\mathcal{E}(\rho)$, then you simply follow Equation (\natural) , taking the values of $\mathcal{E}(|i\rangle\langle j|)$ from the Choi matrix. We can write this more formally as follows.

The Choi matrix $\tilde{\mathcal{E}}$ of a linear map $\mathcal{E}: \mathcal{B}(\mathcal{H}) \rightarrow \mathcal{B}(\mathcal{H}')$ satisfies

$$\frac{1}{d}\mathcal{E}(\rho) = (\text{tr} \otimes \mathbf{1}) \left[(\rho^T \otimes \mathbf{1}_{d' \times d'}) \tilde{\mathcal{E}} \right]$$

for all density matrices ρ in $\mathcal{B}(\mathcal{H})$, where $d = \dim \mathcal{H}$.

Man-Duen Choi was brought up in Hong Kong. He received his Ph.D. degree under the guidance of Chandler Davis at Toronto. He taught at the University of California, Berkeley, from 1973 to 1976, and has worked since then at the University of Toronto. His research has been mainly in operator algebras, operator theory, and polynomial rings. He is particularly interested in examples/counterexamples and 2×2 matrix manipulations.

The expression above may look baffling at first glance, but this is often the case when we turn something conceptually obvious into more compact mathematical notation. In order to gain some intuition here, recall that, for matrices A and B ,

$$\text{tr } A^T B = \sum_{i,j} A_{ij} B_{ij}.$$

If we take A and B to be the block matrices $\rho \otimes \mathbf{1}$ and $\tilde{\mathcal{E}}$, respectively, then we can use this to show that

$$(\text{tr} \otimes \mathbf{1}) \left[(\rho^T \otimes \mathbf{1}) \tilde{\mathcal{E}} \right] = \frac{1}{d} \sum_{i,j} \rho_{ij} \mathcal{E}(|i\rangle\langle j|).$$

This gives us a one-to-one correspondence between linear maps $\mathcal{E}: \mathcal{B}(\mathcal{H}) \rightarrow \mathcal{B}(\mathcal{H}')$ and matrices $\tilde{\mathcal{E}}$ acting on the tensor product $\mathcal{H} \otimes \mathcal{H}'$, known as the **Choi–Jamiokowski isomorphism** $\mathcal{E} \mapsto \tilde{\mathcal{E}}$.

The Choi–Jamiokowski isomorphism.

The correspondence between linear maps $\mathcal{B}(\mathcal{H}) \rightarrow \mathcal{B}(\mathcal{H}')$ and operators in $\mathcal{B}(\mathcal{H} \otimes \mathcal{H}')$, known as the **Choi–Jamiokowski isomorphism** (or **channel-state duality** in the specific setting of quantum information), is another example of a well known correspondence between vectors in $\mathcal{H}_A \otimes \mathcal{H}_B$ and operators $\mathcal{B}(\mathcal{H}_A^*, \mathcal{H}_B)$ or $\mathcal{B}(\mathcal{H}_B^*, \mathcal{H}_A)$.

Take a tensor product vector in $|a\rangle\otimes|b\rangle \in \mathcal{H}_A\otimes\mathcal{H}_B$. Then it defines natural maps in $\mathcal{B}(\mathcal{H}_A^*, \mathcal{H}_B)$ and $\mathcal{B}(\mathcal{H}_B^*, \mathcal{H}_A)$, via

$$\begin{aligned}\langle x| &\longmapsto \langle x|a\rangle|b\rangle \\ \langle y| &\longmapsto |a\rangle\langle y|b\rangle\end{aligned}$$

for any linear forms $\langle x| \in \mathcal{H}_A^*$ and $\langle y| \in \mathcal{H}_B^*$. We then extend this construction (by linearity) to any vector in $\mathcal{H}_A \otimes \mathcal{H}_B$. These isomorphisms are **canonical**: they do not depend on the choice of any bases in the vectors spaces involved.

However, some care must be taken when we want to define correspondence between vectors in $\mathcal{H}_A \otimes \mathcal{H}_B$ and operators in $\mathcal{B}(\mathcal{H}_A, \mathcal{H}_B)$ or $\mathcal{B}(\mathcal{H}_B, \mathcal{H}_A)$. For example, physicists like to “construct” $\mathcal{B}(\mathcal{H}_B, \mathcal{H}_A)$ in a deceptively simple way:

$$|a\rangle|b\rangle \longleftrightarrow |a\rangle\langle b|.$$

Flipping $|b\rangle$ and switching from \mathcal{H}_B to \mathcal{H}_B^* is an *anti-linear* operation (since it involves complex conjugation). This is fine *when we stick to a specific basis* $|i\rangle|j\rangle$ and use the ket-flipping approach *only for the basis vectors*. This means that, for $|b\rangle = \sum_j \beta_j |j\rangle$, the correspondence looks like

$$|i\rangle|b\rangle \longleftrightarrow \sum_j \beta_j |i\rangle\langle j|$$

and *not* like

$$|i\rangle|b\rangle \longleftrightarrow |i\rangle\langle b| = \sum_j \beta_j^* |i\rangle\langle j|.$$

This isomorphism is **non-canonical**: it depends on the choice of the basis. But it is still a pretty useful isomorphism! The Choi–Jamiokowski isomorphism is of this kind (i.e. non-canonical) — it works in the basis in which you express a maximally mixed state $|\Omega\rangle = \sum_i |i\rangle|i\rangle$.

Mathematically, it is not too surprising that the matrix elements of an operator on a tensor product can be reorganised and reinterpreted as the matrix elements of an operator between operator spaces. What is interesting, and perhaps not so obvious, however, is that the positivity conditions for maps correspond exactly to conditions on their Choi matrices under this correspondence. That is, this one-to-one correspondence between linear maps $\mathcal{E}: \mathcal{B}(\mathcal{H}) \rightarrow \mathcal{B}(\mathcal{H}')$ and matrices $\tilde{\mathcal{E}}$ acting on the tensor product $\mathcal{H} \otimes \mathcal{H}'$ descends to a one-to-one correspondence between quantum channels and some specific family of matrices (which we will shortly discuss). In other words, we can classify quantum channels as being exactly those linear maps that have a certain image under the Choi–Jamiokowski isomorphism! In order to see this, let us express the Choi matrix as the result of $\mathbf{1} \otimes \mathcal{E}$ acting on the maximally mixed state

$$|\Omega\rangle := \frac{1}{\sqrt{d}} \sum_{i=1}^d |i\rangle|i\rangle$$

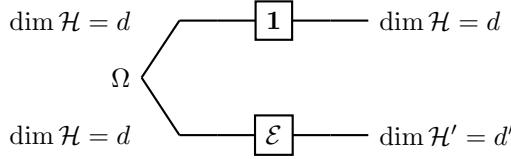
in $\mathcal{H} \otimes \mathcal{H}$.

The Choi matrix $\tilde{\mathcal{E}}$ of a linear map $\mathcal{E}: \mathcal{B}(\mathcal{H}) \rightarrow \mathcal{B}(\mathcal{H}')$ is given by

$$\tilde{\mathcal{E}} = (\mathbf{1}_{d \times d} \otimes \mathcal{E})|\Omega\rangle\langle\Omega| = \frac{1}{d} \sum_{i,j} |i\rangle\langle j| \otimes \mathcal{E}(|i\rangle\langle j|)$$

where $d = \dim \mathcal{H}$.

Pictorially, we might represent this by something like



In this form, we can see right away that, if \mathcal{E} is a quantum channel, then $\tilde{\mathcal{E}}$ is a density matrix. In fact, not just any density matrix: the first subsystem of the maximally entangled state $|\Omega\rangle$ is initially maximally, and remains maximally mixed, since we apply the identity operator, and so $(\mathbf{1} \otimes \text{tr})\tilde{\mathcal{E}} = \frac{1}{d}\mathbf{1}$. The converse is also true: any density matrix $\tilde{\mathcal{E}}$ such that $(\mathbf{1} \otimes \text{tr})\tilde{\mathcal{E}} = \frac{1}{d}\mathbf{1}$ defines a quantum channel, i.e. a completely positive trace-preserving map. This is just one example of how, in general, the Choi-Jamiolkowski isomorphism provides a simple way of studying linear maps on operators by means of inspecting their Choi matrices.

Let $\tilde{\mathcal{E}}$ be the Choi matrix of a linear map $\mathcal{E}: \mathcal{B}(\mathcal{H}) \rightarrow \mathcal{B}(\mathcal{H}')$. Then

1. \mathcal{E} is completely positive if and only if $\tilde{\mathcal{E}}$ is positive semi-definite.
2. \mathcal{E} is trace preserving if and only if $(\mathbf{1} \otimes \text{tr})\tilde{\mathcal{E}} = \frac{1}{d}\mathbf{1}$.
3. \mathcal{E} sends the identity operator to the identity operator if and only if $(\text{tr} \otimes \mathbf{1})\tilde{\mathcal{E}} = \frac{1}{d}\mathbf{1}$.
4. \mathcal{E} sends Hermitian operators to Hermitian operators if and only if $\tilde{\mathcal{E}}$ is Hermitian.

We shall prove the first two of these correspondences here, and leave the last two as an exercise.

Let's start with complete positivity, since one direction is much easier: if \mathcal{E} is a completely positive map, then its extension $\mathbf{1} \otimes \mathcal{E}$ maps $|\Omega\rangle\langle\Omega|$ to a positive semi-definite matrix, and so $\tilde{\mathcal{E}}$ is positive semi-definite. The converse is less immediate. If $\tilde{\mathcal{E}}$ is positive semi-definite, then its eigenvalues p_k are non-negative, and we can write its spectral decomposition as

$$\tilde{\mathcal{E}} = \sum_k p_k |\psi_k\rangle\langle\psi_k| = \sum_k |\tilde{\psi}_k\rangle\langle\tilde{\psi}_k|$$

where the vectors $|\tilde{\psi}_k\rangle = \sqrt{p_k}|\psi_k\rangle$ are pairwise orthogonal but not normalised. Each of the vectors $|\tilde{\psi}_k\rangle$ can be written as

$$|\tilde{\psi}_k\rangle = (\mathbf{1} \otimes E_k)|\Omega\rangle$$

for some operator E_k (Exercise 9.12.15). This means that

$$\begin{aligned}\tilde{\mathcal{E}} &= \sum_k |\tilde{\psi}_k\rangle\langle\tilde{\psi}_k| \\ &= \sum_k (\mathbf{1} \otimes E_k) |\Omega\rangle\langle\Omega| (\mathbf{1} \otimes E_k^\dagger) \\ &= \frac{1}{d} \sum_{i,j} \left(|i\rangle\langle j| \otimes \underbrace{\sum_k E_k \langle i|j\rangle E_k^\dagger}_{\mathcal{E}(|i\rangle\langle j|)} \right).\end{aligned}$$

Comparing this last expression with the definition of $\tilde{\mathcal{E}}$, we conclude that \mathcal{E} is of the form

$$\mathcal{E}(\rho) = \sum_k E_k \rho E_k^\dagger$$

which is a completely positive map in Kraus form (though not necessarily trace preserving, since we do not require that $\sum_k E_k E_k^\dagger = \mathbf{1}$).

For the trace-preserving correspondence, note first of all that, if \mathcal{E} is trace preserving, then

$$\begin{aligned}(\mathbf{1} \otimes \text{tr})\tilde{\mathcal{E}} &= \frac{1}{d} \sum_{i,j} |i\rangle\langle j| \underbrace{\text{tr } \mathcal{E}(|i\rangle\langle j|)}_{\delta_{ij}} \\ &= \frac{1}{d} \sum_i |i\rangle\langle i| \\ &= \frac{1}{d} \mathbf{1}.\end{aligned}$$

Conversely, for any operator ρ in $\mathcal{B}(\mathcal{H})$, we have already seen that

$$\frac{1}{d} \text{tr } \mathcal{E}(\rho) = (\text{tr} \otimes \mathbf{1}) \left[(\rho^T \otimes \mathbf{1}) \tilde{\mathcal{E}} \right]$$

and so, tracing over \mathcal{H}' by applying $\mathbf{1} \otimes \text{tr}$, we see that

$$\text{tr } \mathcal{E}(\rho) = (\mathbf{1} \otimes \text{tr}) \left[d(\text{tr} \otimes \mathbf{1}) \left[(\rho^T \otimes \mathbf{1}) \tilde{\mathcal{E}} \right] \right]$$

which rearranges to give

$$\begin{aligned}\text{tr } \mathcal{E}(\rho) &= d(\text{tr} \otimes \text{tr}) \left[(\rho^T \otimes \mathbf{1}) \tilde{\mathcal{E}} \right] \\ &= d(\text{tr} \otimes \mathbf{1}) \left[\rho^T (\mathbf{1} \otimes \text{tr}) \tilde{\mathcal{E}} \right]\end{aligned}$$

by using the fact that $(\text{tr} \otimes \text{tr})[(A \otimes \mathbf{1})C] = (\text{tr} \otimes \mathbf{1})[A(\mathbf{1} \otimes \text{tr})C]$, which is just another way of writing the defining property of the partial trace: $\text{tr}_{\mathcal{A}\mathcal{B}}(A \otimes \mathbf{1})C = \text{tr}_{\mathcal{A}}(A \text{tr}_{\mathcal{B}} C)$. So if $(\mathbf{1} \otimes \text{tr})\tilde{\mathcal{E}} = \frac{1}{d}\mathbf{1}$, then

$$\text{tr } \mathcal{E}(\rho) = \text{tr } \rho^T = \text{tr } \rho$$

and so \mathcal{E} is trace preserving. Note that we have already used this defining property of the partial trace when calculating the expectation value of an observable A that pertains only to a subsystem \mathcal{A} of a bipartite system $\mathcal{A}\mathcal{B}$ described by some density operator $\rho_{\mathcal{A}\mathcal{B}}$, noting that $\text{tr}[(A \otimes \mathbf{1})\rho_{\mathcal{A}\mathcal{B}}] = \text{tr}[A\rho_{\mathcal{A}}]$, where $\rho_{\mathcal{A}} = \text{tr}_{\mathcal{B}} \rho_{\mathcal{A}\mathcal{B}}$.

In particular then, completely positive trace-preserving maps (quantum channels) have Choi matrices that are positive semi-definite and such that their partial trace

gives the maximally mixed state $\frac{1}{d}\mathbf{1}$, and we have just shown that the converse is true.

Channel-state duality. The following three things are all equivalent to one another:

- quantum channels (i.e. linear maps that can be written in Stinespring or Kraus form)
- completely positive trace-preserving (CPTP) maps
- linear maps \mathcal{E} whose Choi matrix $\tilde{\mathcal{E}}$ is positive semi-definite and such that $(\mathbf{1} \otimes \text{tr})\tilde{\mathcal{E}} = \frac{1}{d}\mathbf{1}$.

Furthermore, all completely positive maps admit a Kraus decomposition $\rho \mapsto \sum_k E_k \rho E_k^\dagger$, and these Kraus operators can be obtained from the spectral decomposition of the corresponding Choi matrix. Given the Kraus decomposition, if we *also* want the map to be trace preserving, then we must additionally require that the Kraus operators satisfy $\sum_k E_k^\dagger E_k = \mathbf{1}$.

9.10 The mathematics of “can” and “cannot”

So what is channel-state duality good for? To start with, it can be used to assess whether or not a given map $\mathcal{B}(\mathcal{H}) \rightarrow \mathcal{B}(\mathcal{H}')$ can actually be physically implemented, i.e. if it is a CPTP map. Indeed, all we have to do is to check if the corresponding Choi matrix is a density matrix. Let’s look at a simple example.

Consider the map

$$\mathcal{E}: |i\rangle\langle j| \mapsto p|j\rangle\langle i| + (1-p)\delta_{ij}\frac{1}{2}\mathbf{1}$$

Again, δ_{ij} is the Kronecker delta, which is equal to 1 if $i = j$ and equal to 0 if $i \neq j$.

where $0 \leq p \leq 1$ is some fixed parameter. This map acts on a density operator ρ via

$$\rho \mapsto p\rho^T + (1-p)\frac{1}{2}\mathbf{1}$$

(where ρ^T is the transpose of ρ).

But is this map a quantum channel? That is, *does it represent a physical process that can be implemented in a lab?*

We can interpret the convex-sum expression

$$\mathcal{E}(|i\rangle\langle j|) = p|j\rangle\langle i| + (1-p)\delta_{ij}\frac{1}{2}|i\rangle\langle j|$$

as follows: take the input state ρ and either (i) apply the transpose, with probability p , or (ii) replace it with the maximally mixed state, with probability $1 - p$. This is fine, except that the transpose operation is *not* completely positive, and, as such, is *not physically admissible* — it cannot be implemented. But does this mean that the map \mathcal{E} itself cannot be implemented? Not necessarily!

In fact, the answer depends on the value of p . The case $p = 0$ corresponds to just replacing the input with the maximally mixed state, which is something that can be easily implemented. However, as p increases from 0 to 1, at some critical point the map switches from *completely positive* to merely *positive*. In order to find this critical

value of p , we first calculate $\mathcal{E}(|i\rangle\langle j|)$ for $i, j \in \{0, 1\}$ as follows:

$$\begin{aligned} |0\rangle\langle 0| &= \begin{bmatrix} 1 & 0 \\ 0 & 0 \end{bmatrix} \xrightarrow{\mathcal{E}} \begin{bmatrix} \frac{1+p}{2} & 0 \\ 0 & \frac{1-p}{2} \end{bmatrix} \\ |0\rangle\langle 1| &= \begin{bmatrix} 0 & 1 \\ 0 & 0 \end{bmatrix} \xrightarrow{\mathcal{E}} \begin{bmatrix} 0 & 0 \\ p & 0 \end{bmatrix}, \\ |1\rangle\langle 0| &= \begin{bmatrix} 0 & 0 \\ 1 & 0 \end{bmatrix} \xrightarrow{\mathcal{E}} \begin{bmatrix} 0 & p \\ 0 & 0 \end{bmatrix} \\ |1\rangle\langle 1| &= \begin{bmatrix} 0 & 0 \\ 0 & 1 \end{bmatrix} \xrightarrow{\mathcal{E}} \begin{bmatrix} \frac{1-p}{2} & 0 \\ 0 & \frac{1+p}{2} \end{bmatrix}, \end{aligned}$$

We can then write down the Choi matrix:

$$\tilde{\mathcal{E}} = \frac{1}{2} \begin{bmatrix} \mathcal{E}(|0\rangle\langle 0|) & \mathcal{E}(|0\rangle\langle 1|) \\ \mathcal{E}(|1\rangle\langle 0|) & \mathcal{E}(|1\rangle\langle 1|) \end{bmatrix} = \frac{1}{2} \left[\begin{array}{cc|cc} \frac{1+p}{2} & 0 & 0 & 0 \\ 0 & \frac{1-p}{2} & p & 0 \\ \hline 0 & p & \frac{1-p}{2} & 0 \\ 0 & 0 & 0 & \frac{1+p}{2} \end{array} \right]$$

which lets us apply channel-state duality: \mathcal{E} is completely positive (and hence physically realisable) if and only if $\tilde{\mathcal{E}} \geq 0$, and the latter is true only when $p \leq \frac{1}{3}$ (note that the eigenvalues of $\tilde{\mathcal{E}}$ are $\frac{1}{4}(1+p)$ and $\frac{1}{4}(1-3p)$).

9.11 Kraus operators, revisited

Channel-state duality gives us more than just a one-to-one correspondence between states $\tilde{\mathcal{E}}$ and channels \mathcal{E} — it also gives a one-to-one correspondence between vectors in the statistical ensemble $\tilde{\mathcal{E}}$ and the Kraus operators in the decomposition of \mathcal{E} .

With the above in mind, we see that the freedom to choose the Kraus operators representing a channel in many different ways is really the same thing as the freedom to choose the ensemble of pure states representing a density operator in many different ways.

We already know that if two mixtures $(p_k, |\psi_k\rangle)$ and $(q_l, |\phi_l\rangle)$ are described by the same density operator

$$\sum_k |\tilde{\psi}_k\rangle\langle\tilde{\psi}_k| = \tilde{\mathcal{E}} = \sum_l |\tilde{\phi}_l\rangle\langle\tilde{\phi}_l|$$

(where $|\tilde{\psi}_k\rangle = \sqrt{p_k}|\psi_k\rangle$ and $|\tilde{\phi}_l\rangle = \sqrt{q_l}|\phi_l\rangle$) then they are related to one another: there exists some unitary R such that

$$|\tilde{\psi}_k\rangle = \sum_l R_{kl} |\tilde{\phi}_l\rangle.$$

Using the aforementioned fact that any vector $|\psi\rangle$ in $\mathcal{H} \otimes \mathcal{H}'$ can be written as $|\psi\rangle = \mathbf{1} \otimes V|\Omega\rangle$, this implies the same unitary freedom in choosing the Kraus operators.

So how many Kraus operators do we really need? Channel-state duality tells us that the minimal number of Kraus operators needed to express $\mathcal{E}: \mathcal{B}(\mathcal{H}) \rightarrow \mathcal{B}(\mathcal{H}')$ in the operator-sum form is given by the rank of its Choi matrix $\tilde{\mathcal{E}}$, i.e. we need no more than dd' such operators (where $d = \dim \mathcal{H}$ and $d' = \dim \mathcal{H}'$). In fact, this minimal set of Kraus operators corresponds to the spectral decomposition of $\tilde{\mathcal{E}}$.

Indeed, if $\tilde{\mathcal{E}} = \sum_k |\tilde{v}_k\rangle\langle\tilde{v}_k|$ and $|\tilde{v}_k\rangle = (\mathbf{1} \otimes E_k)|\Omega\rangle$, then the orthogonality of $|\tilde{v}_k\rangle$ and $|\tilde{v}_l\rangle$ implies the orthogonality (in the Hilbert–Schmidt sense) of the corresponding

The number of vectors contributing to each mixture (and hence the number of corresponding Kraus operators) may be different, but we can always simply extend the smaller set to the required size by adding zero operators.

We talk about spectral decomposition in more detail in Section 12.11.1.

Recall that the Hilbert–Schmidt product $(A|B)$ of two operators A and B is defined by $(A|B) = \frac{1}{2} \text{tr } A^\dagger B$.

Kraus operators E_k and E_l . In order to see this, we write $\langle \tilde{v}_k | \tilde{v}_l \rangle$ as

$$\begin{aligned}\langle \tilde{v}_k | \tilde{v}_l | \tilde{v}_k | \tilde{v}_l \rangle &= \langle \Omega | (\mathbf{1} \otimes E_k^\dagger)(\mathbf{1} \otimes E_l) | \Omega \rangle \\ &= \text{tr}(\mathbf{1} \otimes E_k^\dagger E_l) | \Omega \rangle \langle \Omega | \\ &= \frac{1}{d} \text{tr} \sum_{i,j} |i\rangle \langle j| \otimes E_k^\dagger E_l |i\rangle \langle j|\end{aligned}$$

(using the fact that we can substitute $\frac{1}{d} \sum_{i,j} |i\rangle \langle j| \otimes |i\rangle \langle j|$ for $|\Omega\rangle \langle \Omega|$). Now, the trace of the tensor product of two matrices is the product of their traces, hence

$$\begin{aligned}\langle \tilde{v}_k | \tilde{v}_l \rangle &= \frac{1}{d} \sum_{i,j} \langle i | j \rangle \text{tr} E_k^\dagger E_l |i\rangle \langle j| \\ &= \frac{1}{d} \text{tr} E_k^\dagger E_l\end{aligned}$$

(using the fact that $\langle i | j \rangle = \delta_{ij}$ and $\sum_i |i\rangle \langle i| = \mathbf{1}$). So we have shown that if $\langle \tilde{v}_k | \tilde{v}_l \rangle = 0$ then $\text{tr} E_k^\dagger E_l = 0$.

A linear map $\mathcal{E}: \mathcal{B}(\mathcal{H}) \rightarrow \mathcal{B}(\mathcal{H}')$ is completely positive if and only if it admits an operator-sum decomposition of the form

$$\mathcal{E}(\rho) = \sum_k E_k \rho E_k^\dagger.$$

If this is the case, then this decomposition has the following properties:

- \mathcal{E} is trace preserving if and only if $\sum_k E_k^\dagger E_k = \mathbf{1}$.
- Two sets of Kraus operators $\{E_k\}$ and $\{F_l\}$ represent the same map \mathcal{E} if and only if there exists a unitary R such that $E_k = \sum_l R_{kl} F_l$ (where the smaller set of the Kraus operators is padded with zeros, if necessary).

Note that, for any $\mathcal{E}: \mathcal{B}(\mathcal{H}) \rightarrow \mathcal{B}(\mathcal{H}')$, there always exists a representation with at most dd' mutually orthogonal Kraus operators: $\text{tr} E_i^\dagger E_j \propto \delta_{ij}$.

For example, consider the simpler case where $\dim \mathcal{H} = \dim \mathcal{H}' = d$. Then the Kraus operators E_k are vectors in a d^2 -dimensional Hilbert space, with the Hilbert–Schmidt inner product $\text{tr} E_k^\dagger E_l$. We can pick an orthonormal basis of operators $\{B_i\}$ and express each Kraus vector in this basis as $E_k = \sum_i c_{ki} B_i$ (where $i = 1, \dots, d^2$ and $k = 1, \dots, n$, with n possibly much larger than d^2). This gives us

$$\begin{aligned}\rho &\mapsto \sum_{i,j} B_i \rho B_j^\dagger \left(\sum_k c_{ki} c_{kj}^* \right) \\ &= \sum_{i,j} B_i \rho B_j^\dagger C_{ij}\end{aligned}$$

The matrix C_{ij} is positive semi-definite, and hence unitarily diagonalisable: $C_{ij} = \sum_k U_{ik} d_k U_{kj}^\dagger$ for some unitary U and some $d_k \geq 0$. We can then unitarily “rotate” our operator basis and use the $C_k = \sum_j U_{jk} B_j \sqrt{d_k}$ as our new Kraus operators.

The utility of Kraus operators when it comes to understanding quantum channels will be even more obvious when we prove some facts about correctable channels in Section ??.

9.12 Remarks and exercises

9.12.1 Purifications and isometries

All purifications of a density operator are related by an isometry acting on the purifying system. That is, if ρ is a density operator on \mathcal{H} , and $|\psi_A\rangle \in \mathcal{H} \otimes \mathcal{H}_A$ and $|\psi_B\rangle \in \mathcal{H} \otimes \mathcal{H}_B$ are two purifications of ρ with $\dim \mathcal{H}_A \leq \dim \mathcal{H}_B$, then

$$|\psi_B\rangle = (\mathbf{1} \otimes V)|\psi_A\rangle$$

for some isometry V .

To show this, we start with the spectral decomposition of ρ

$$\rho = \sum_i p_i |i\rangle\langle i|$$

and note that

$$|\psi_A\rangle = \sum_i \sqrt{p_i} |i\rangle \otimes |a_i\rangle$$

$$|\psi_B\rangle = \sum_i \sqrt{p_i} |i\rangle \otimes |b_i\rangle$$

which defines an isometry $V = \sum_i |b_i\rangle\langle a_i|$ satisfying the desired equation.

This observation leads to a way of relating *all* convex decompositions of a given density operator: let $(p_k, |\psi_k\rangle)$ and $(q_l, |\phi_l\rangle)$ be convex decompositions of a density operator ρ ; then there exists an isometry V such that these two decompositions

$$\sum_{k=1}^n |\tilde{\psi}_k\rangle\langle\tilde{\psi}_k| = \rho = \sum_{l=1}^m |\tilde{\phi}_l\rangle\langle\tilde{\phi}_l|$$

(where $n \geq m$, and $|\tilde{\psi}_k\rangle = \sqrt{p_k}|\psi_k\rangle$ and $|\tilde{\phi}_l\rangle = \sqrt{q_l}|\phi_l\rangle$) are related:

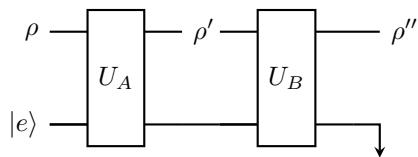
$$|\tilde{\psi}_k\rangle = \sum_l V_{kl} |\tilde{\phi}_l\rangle.$$

9.12.2 The Markov approximation

Unitary evolutions form a group, but quantum channels form a [semigroup](#), since they are not necessarily invertible. Indeed, quantum operations are invertible only if they are either unitary operations or simple isometric embeddings (such as the process of bringing in the environment in some fixed state and then *immediately* discarding it, without any intermediate interaction).

Anyway, composition of quantum channels in the Kraus representation is rather straightforward, but do not be deceived by its mathematical simplicity! We must remember that *quantum channels do not capture all possible quantum evolutions*: the assumption that the system and the environment are *not initially correlated* is crucial, and it does impose some restrictions on the applicability of our formalism. Compare, for example, the following two scenarios.

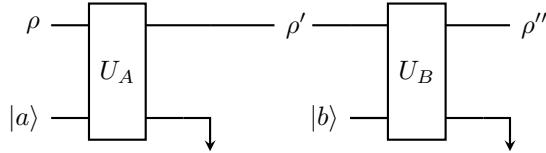
Firstly:



Here we have reverted to the convention of writing the ancilla/environment *after* the system of interest instead of *before*.

Here the system, initially in state ρ , undergoes two stages of evolution, and the environment, initially in state $|e\rangle$, is *not* discarded after the first unitary evolution U_A ; the environment persists and participates in the second unitary evolution U_B . In this case the evolutions $\rho \mapsto \rho'$ and $\rho \mapsto \rho''$ are both well defined quantum channels, *but the evolution $\rho' \mapsto \rho''$ is not*: it falls outside the remit of our formalism because the input state of the system and the state of the environment are *not independent*.

Secondly:



Here we have two stages of evolution, as before, but we *discard* the environment after the first unitary, and start the second unitary evolution in a fresh tensor-product state, with a *new* environment; the two stages involve *independent environments*. In this case all three evolutions ($\rho \mapsto \rho'$, $\rho' \mapsto \rho''$, and $\rho \mapsto \rho''$) are well defined quantum channels, and they compose: if \mathcal{E}_A describes the evolution from ρ to ρ' , and \mathcal{E}_B from ρ' to ρ'' , then the composition $\mathcal{E}_B \circ \mathcal{E}_A$ describes the evolution from ρ to ρ'' .

In practice we often deal with complex environments that have internal dynamics that “hides” any entanglement with the system as quickly as it arises. For example, suppose that our system is an atom, surrounded by the electromagnetic field (which serves as the environment). Let the field start in the vacuum state. If the atom emits a photon into the environment, then the photon quickly propagates away, and the immediate vicinity of the atom appears to be empty, i.e. resets to the vacuum state. In this approximate model, we assume that the environment quickly forgets about the state resulting from any previous evolution. This is known as the **Markov approximation** — in a quantum Markov process the environment has essentially no memory.

9.12.3 What use are positive maps?

Positive maps that are not completely positive are not completely useless. True, they cannot describe any quantum dynamics, but still they have useful applications — for example, they can help us to determine if a given state is entangled or not.

Recall that a quantum state of a bipartite system \mathcal{AB} described by the density matrix $\varrho_{\mathcal{AB}}$ is said to be **separable** if $\varrho_{\mathcal{AB}}$ can be written in the form

$$\varrho_{\mathcal{AB}} = \sum_k p_k \rho_{\mathcal{A},k} \otimes \rho_{\mathcal{B},k}$$

where $\rho_{\mathcal{A},k}$ are density matrices on \mathcal{A} and $\rho_{\mathcal{B},k}$ are density matrices on \mathcal{B} (and where $p_k \geq 0$ and $\sum_k p_k = 1$); otherwise $\varrho_{\mathcal{AB}}$ is said to be **entangled**. If we apply the partial transpose $\mathbf{1} \otimes T$ to this state, then it remains separable, since, as we have seen, the transpose ρ^B is a legal density matrix.

In separable states, one subsystem does not really know about the existence of the other, and so applying a positive map to one part produces a proper density operator, and thus does *not* reveal the unphysical character of the map. So, for *any separable state* ρ , we have $(\mathbf{1} \otimes T)\rho \geq 0$.

Positive (but not completely positive) maps, such as the transpose, can be quite deceptive: you have to include other systems in order to detect their unphysical character.

In particular, positive maps appear to be completely positive *on separable states*.

A **quantum Markov process!**
 Andrey Markov (1929–2012) was a Russian mathematician best known for his work on stochastic processes.

As an example, consider a quantum state ρ_p of two qubits which is a mixture of the maximally mixed state $|\Omega\rangle = \frac{1}{\sqrt{2}}(|00\rangle + |11\rangle)$ and the identity matrix with respective probabilities p and $1 - p$. That is,

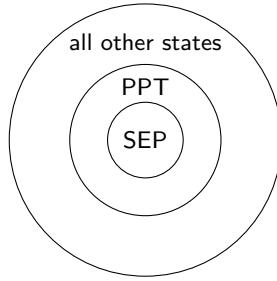
$$\rho_p = p|\Omega\rangle\langle\Omega| + \frac{(1-p)}{4}\mathbf{1} \otimes \mathbf{1}.$$

If we apply the partial transpose $\mathbf{1} \otimes T$ to this state, and check for which values of p the resulting matrix is a density matrix, we can show that the density operator ρ_p describes an entangled state for all $p \in [\frac{1}{3}, 1]$.

We say that a state is a **PPT state** if its partial transpose is positive. An important thing to note is that separable states are PTT, but the converse is generally *not* true: there exist entangled PPT states. However, in the specific case of two qubits, the converse *is* true: the PPT states are exactly the separable states.

Recall that a state is said to be **maximally mixed** if the outcomes of *any* measurement on that state are completely random.

“PPT” stands for *positive partial transpose*.



9.12.4 Partial inner product

Tensor products bring the possibility to do “partial things” beyond just the partial trace. Given $\mathcal{H}_A \otimes \mathcal{H}_B$, any vector $|x\rangle \in \mathcal{H}_A$ defines an anti-linear map $\mathcal{H}_A \otimes \mathcal{H}_B \rightarrow \mathcal{H}_B$ called the **partial inner product with $|x\rangle$** . It is first defined on the product vectors $|a\rangle \otimes |b\rangle$ by the formula

$$|a\rangle \otimes |b\rangle \longmapsto \langle x|a\rangle|b\rangle$$

and then extended to other vectors in $\mathcal{H}_A \otimes \mathcal{H}_B$ by linearity. Similarly, any $|y\rangle \in \mathcal{H}_B$ defines a map $\mathcal{H}_A \otimes \mathcal{H}_B \rightarrow \mathcal{H}_A$ via

$$|a\rangle \otimes |b\rangle \longmapsto |a\rangle\langle y|b\rangle$$

For example, the partial inner product of

$$|\psi\rangle = c_{00}|00\rangle + c_{01}|01\rangle + c_{10}|10\rangle + c_{11}|11\rangle \in \mathcal{H}_A \otimes \mathcal{H}_B$$

with $|0\rangle \in \mathcal{H}_A$ is

$$\langle 0|\psi\rangle = c_{00}|0\rangle + c_{01}|1\rangle$$

and the partial inner product of the same $|\psi\rangle$ with $|1\rangle \in \mathcal{H}_B$ is

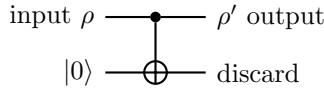
$$\langle 1|\psi\rangle = c_{01}|0\rangle + c_{11}|1\rangle.$$

9.12.5 The “control” part of controlled-NOT

Consider a single-qubit channel induced by the action of the c-NOT gate. Recall that the unitary operator associated with the c-NOT gate can be written as

$$U = |0\rangle\langle 0| \otimes \mathbf{1} + |1\rangle\langle 1| \otimes X$$

where X is the Pauli σ_x gate (i.e. the NOT gate). Let us step through the following simple circuit:



This time we are interested in the evolution of the *control* qubit: the control qubit will be our system, and the target qubit, initially in a fixed state $|0\rangle$, will play the role of an ancilla.

We can calculate the Kraus operators:

$$E_i = (\mathbf{1} \otimes \langle i|)U(\mathbf{1} \otimes |0\rangle)$$

which we simply write as $E_i = \langle i|U|0\rangle$ (for $i = 0, 1$). Expanding out the definition of U , we see that

$$\begin{aligned} E_i = \langle i|U|0\rangle &= \langle i|(|0\rangle\langle 0| \otimes \mathbf{1} + |1\rangle\langle 1| \otimes X)|0\rangle \\ &= |0\rangle\langle 0|i|1|0\rangle + |1\rangle\langle 1|i|X|0\rangle \\ &= |i\rangle\langle i| \end{aligned}$$

We can also check the normalisation condition:

$$E_0^\dagger E_0 + E_1^\dagger E_1 = |0\rangle\langle 0| + |1\rangle\langle 1| = \mathbf{1}.$$

The unitary action of the gate when the state of the target qubit is fixed at $|0\rangle$ can be written as

$$\begin{aligned} |\psi\rangle|0\rangle &\mapsto E_0|\psi\rangle|0\rangle + E_1|\psi\rangle|1\rangle \\ &= |0\rangle\langle 0||\psi\rangle|0\rangle + |1\rangle\langle 1||\psi\rangle|1\rangle \\ &= \langle 0|\psi\rangle|0\rangle|0\rangle + \langle 1|\psi\rangle|1\rangle|1\rangle \end{aligned}$$

which is a familiar c-NOT entangling process: if $|\psi\rangle = \alpha_0|0\rangle + \alpha_1|1\rangle$ then $|\psi\rangle|0\rangle$ evolves into $\alpha_0|0\rangle|0\rangle + \alpha_1|1\rangle|1\rangle$.

The evolution of the control qubit alone can be expressed in the Kraus form as

$$\begin{aligned} \rho &\mapsto \rho' = E_0\rho E_0^\dagger + E_1\rho E_1^\dagger \\ &= |0\rangle\langle 0|\rho|0\rangle\langle 0| + |1\rangle\langle 1|\rho|1\rangle\langle 1| \\ &= \rho_{00}|0\rangle\langle 0| + \rho_{11}|1\rangle\langle 1|. \end{aligned}$$

Then, in the matrix form, if the initial state of the control qubit is $|\psi\rangle = \alpha_0|0\rangle + \alpha_1|1\rangle$, we get

$$\begin{bmatrix} |\alpha_0|^2 & \alpha_0\alpha_0^* \\ \alpha_0^*\alpha_1 & |\alpha_1|^2 \end{bmatrix} = \rho \mapsto \rho' = \begin{bmatrix} |\alpha_0|^2 & 0 \\ 0 & |\alpha_1|^2 \end{bmatrix}.$$

As we can see, the diagonal elements of ρ survive, and the off-diagonal elements (the **coherences**) disappear. The two Kraus operators, $E_0 = |0\rangle\langle 0|$ and $E_1 = |1\rangle\langle 1|$, define the measurement in the standard basis, and so you may think about this operation as being equivalent to *measuring the control qubit in the standard basis and then just forgetting the result*.

9.12.6 Surprisingly identical channels

Let us now compare two single qubit-quantum channels: $\mathcal{A}(\rho) = \sum_k A_k \rho A_k^\dagger$, defined by the Kraus operators

$$A_1 = |0\rangle\langle 0| = \begin{bmatrix} 1 & 0 \\ 0 & 0 \end{bmatrix}$$

$$A_2 = |1\rangle\langle 1| = \begin{bmatrix} 0 & 0 \\ 0 & 1 \end{bmatrix}$$

and $\mathcal{B}(\rho) = \sum_k B_k \rho B_k^\dagger$, defined by the Kraus operators

$$B_1 = \frac{\mathbf{1}}{\sqrt{2}} = \frac{1}{\sqrt{2}} \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}$$

$$B_2 = \frac{Z}{\sqrt{2}} = \frac{1}{\sqrt{2}} \begin{bmatrix} 1 & 0 \\ 0 & -1 \end{bmatrix}.$$

We are familiar with the first channel from the previous example (9.12.5): it performs the measurement in the standard basis, but *doesn't* reveal the outcome of this measurement. The second channel chooses randomly, with equal probability, between two options: it will either let the qubit pass undisturbed, or apply the phase-flip Z .

These two apparently very different physical processes correspond to the same quantum channel: $\mathcal{A}(\rho) = \mathcal{B}(\rho)$ for any ρ . Indeed, you can check that $B_1 = (A_1 + A_2)/\sqrt{2}$ and $B_2 = (A_1 - A_2)/\sqrt{2}$, whence

$$\begin{aligned} \mathcal{B}(\rho) &= B_1 \rho B_1^\dagger + B_2 \rho B_2^\dagger \\ &= \frac{1}{2}(A_1 + A_2)\rho(A_1 + A_2)^\dagger + \frac{1}{2}(A_1 - A_2)\rho(A_1 - A_2)^\dagger \\ &= A_1 \rho A_1^\dagger + A_2 \rho A_2^\dagger \\ &= \mathcal{A}(\rho). \end{aligned}$$

You can also check that the two channels can be implemented by the following two circuits:

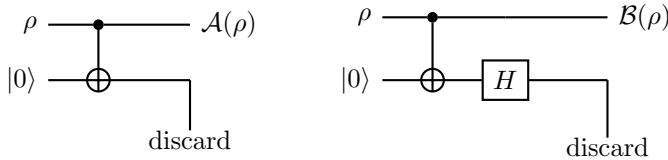


Figure 9.2: The c-NOT gate appears here as the measurement gate. The target qubit (on the bottom) measures the control qubit (on the top) in the standard basis (operation \mathcal{A} on the left) or in the Hadamard basis (operation \mathcal{B} on the right). The extra Hadamard gate on the target qubit has no effect on the control qubit.

9.12.7 Independent ancilla

Another way to understand the freedom in the operator-sum representation is to realise that, once the system and the ancilla cease to interact, any operation on the ancilla alone has no effect on the state of the system.

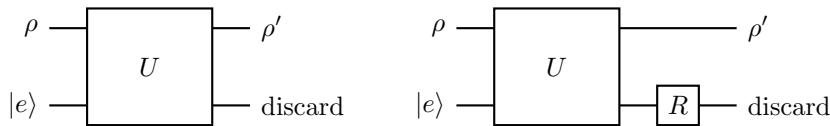


Figure 9.3: The quantum channel $\rho \mapsto \rho'$ is not affected by the choice of a unitary R , and so these two processes are the same.

That is, the two unitaries U and $(\mathbf{1} \otimes R)U$ (where R acts only on the ancilla) describe the same channel, even though the Kraus operators $E_k = \langle e_k | U | e \rangle$ for the

latter are

$$\begin{aligned} F_k &= \langle e_k | (\mathbf{1} \otimes R)U|e\rangle \\ &= \sum_j \langle e_k | R|e_j\rangle \langle e_j | U|e\rangle \\ &= \sum_j R_{kj} E_j \end{aligned}$$

Indeed, the unitary evolution $(\mathbf{1} \otimes R)U$ gives

$$\rho \otimes |e\rangle\langle e| \mapsto \sum_{k,l} E_k \rho E_l^\dagger \otimes R|e_k\rangle\langle e_l|R^\dagger$$

and the subsequent trace over the environment gives

$$\begin{aligned} \text{tr}_E \sum_{k,l} E_k \rho E_l^\dagger \otimes R|e_k\rangle\langle e_l|R^\dagger &= \sum_{k,l} E_k \rho E_l^\dagger \langle e_l | R^\dagger R | e_k \rangle \\ &= \sum_k E_k \rho E_k^\dagger. \end{aligned}$$

9.12.8 Order matters?

We know that, given a fixed state of the environment, the unitaries U and $(\mathbf{1} \otimes R)U$ (where R acts only on the environment) define the same quantum channel. Is the same true for U and $U(\mathbf{1} \otimes R)$ — do these two unitaries define the same quantum channel as one another?

9.12.9 Unchanged reduced density operator

Show that, for any operator ρ on $\mathcal{H}_{\mathcal{A}} \otimes \mathcal{H}_{\mathcal{B}}$ and any operator R on $\mathcal{H}_{\mathcal{B}}$, we have

$$\text{tr}_{\mathcal{B}} [(\mathbf{1} \otimes R)\rho(\mathbf{1} \otimes R^\dagger)] = \text{tr}_{\mathcal{B}} \rho.$$

Hint: show this for separable operators $\rho = A \otimes B$ and then extend the result to any operator ρ by linearity.

That is, the reduced density operator $\rho_{\mathcal{A}} = \text{tr}_{\mathcal{B}} \rho$ is not affected by R .

9.12.10 Cooling down

We can show that the process of cooling a qubit to its ground state, described the map $\mathcal{E}(\rho) = |0\rangle\langle 0|$, is a quantum channel. Indeed, the set of Kraus operators is $|0\rangle\langle 0|$ and $|0\rangle\langle 1|$, and all Bloch vectors are mapped to the Bloch vector representing state $|0\rangle\langle 0|$.

9.12.11 No pancakes

Consider a single-qubit operation which causes the z -component of the Bloch vector to shrink while preserving the values of the x - and y -components. Under such an operation, the Bloch sphere is mapped to an oblate spheroid which touches the Bloch sphere along its equator.

Explain why we cannot physically implement such a map.

9.12.12 Pauli twirl

Show that randomly applying the Pauli operators $\mathbf{1}$, X , Y , and Z , with uniform probability, to any density operator ρ of a single qubit (an operation known as the **Pauli twirl**) results in the maximally mixed state

$$\frac{1}{4}\mathbf{1}\rho\mathbf{1} + \frac{1}{4}X\rho X + \frac{1}{4}Y\rho Y + \frac{1}{4}Z\rho Z = \frac{1}{2}\mathbf{1}.$$

9.12.13 Depolarising channel

The “most popular” Pauli channel is the **depolarising channel**

$$\rho \mapsto (1-p)\rho + \frac{p}{3}(X\rho X + Y\rho Y + Z\rho Z).$$

In the depolarising channel, a qubit in state ρ remains intact with probability $1-p$, or is otherwise transformed with one of the Pauli operators X , Y , and Z , each chosen randomly with probability $p/3$.

Show, using the Pauli twirl (Exercise 9.12.12) or otherwise, that we can rewrite the depolarising channel as

$$\rho \mapsto \rho' = \left(1 - \frac{4}{3}p\right)\rho + \frac{4}{3}p\frac{1}{2}\mathbf{1}.$$

In particular then, we can say that, for $p \leq \frac{3}{4}$, the channel either does nothing or, with probability $\frac{4}{3}p$, throws away the initial quantum state and replaces it by the maximally mixed state.)

It is also instructive to see how the depolarising channel acts on the Bloch sphere. An arbitrary density matrix for a single qubit can be written as

$$\frac{1}{2}(\mathbf{1} + \vec{s} \cdot \vec{\sigma}),$$

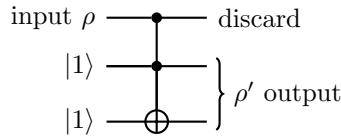
where \vec{s} is the Bloch vector, and $\vec{\sigma} = (\sigma_x, \sigma_y, \sigma_z)$ is the vector of Pauli matrices. The depolarising channel maps this state to

$$\frac{1}{2} \left[\mathbf{1} + \left(1 - \frac{4}{3}p\right) \vec{s} \cdot \vec{\sigma} \right].$$

The Bloch vector shrinks by a factor of $1 - \frac{4}{3}p$. This means that, for $p \leq \frac{3}{4}$, the Bloch sphere contracts uniformly under the action of the channel; for $p = \frac{3}{4}$, the sphere is contracted to a single point at its centre; and for $\frac{3}{4} \leq p \leq 1$, the Bloch vector is flipped, and starts pointing in the opposite direction.

9.12.14 Toffoli gate

Consider the **Toffoli gate**



Express ρ' as a function of ρ in the Kraus representation.

9.12.15 Expressing vectors using the maximally mixed state

Show that any vector $|\psi\rangle$ in $\mathcal{H} \otimes \mathcal{H}'$ can be written as

$$|\psi\rangle = \mathbf{1} \otimes V |\Omega\rangle$$

where $V = \sum_{i,j} V_{ij} |j\rangle\langle i|$ is an operator from \mathcal{H} to \mathcal{H}' , and $|\Omega\rangle = \frac{1}{d} \sum_i |i\rangle|i\rangle$ is a maximally entangled state in $\mathcal{H} \otimes \mathcal{H}$. (Here the vectors $|i\rangle$ and $|j\rangle$ form orthonormal bases in \mathcal{H} and \mathcal{H}' , respectively.)

Recall that a single-qubit Pauli channel is a channel that applies one of the Pauli operators, X , Y or Z , chosen randomly with some prescribed probabilities p_x , p_y and p_z .

9.12.16 Complete positivity of a certain map

Let \mathcal{E} be the linear map on a single qubit defined by

$$\begin{aligned}\mathcal{E}(\mathbf{1}) &= \mathbf{1} \\ \mathcal{E}(\sigma_x) &= a_x \sigma_x \\ \mathcal{E}(\sigma_y) &= a_y \sigma_y \\ \mathcal{E}(\sigma_z) &= a_z \sigma_z\end{aligned}$$

where a_x , a_y , and a_z are some fixed real numbers. Using the Choi matrix of \mathcal{E} , determine the range of a_x , a_y , a_z for which the map \mathcal{E} is *positive*, and the range for which it is *completely positive*.

9.12.17 Duals

We say that $\mathcal{E}^*: \mathcal{B}(\mathcal{H}) \rightarrow \mathcal{B}(\mathcal{H}')$ is the **dual** of a linear map $\mathcal{E}: \mathcal{B}(\mathcal{H}) \rightarrow \mathcal{B}(\mathcal{H}')$ if

$$\text{tr}[\mathcal{E}^*(X)Y] = \text{tr}[X\mathcal{E}(Y)]$$

for any operators X and Y in $\mathcal{B}(\mathcal{H})$.

1. Show that, if \mathcal{E} is trace preserving, then \mathcal{E}^* is **unital** (i.e. that it sends the identity to the identity, or equivalently that its Kraus operators F_j satisfy $\sum_j F_j F_j^\dagger = \mathbf{1}$).
2. Show that, if $\sum_i E_i E_i^\dagger$ is an operator-sum decomposition of \mathcal{E} , then $\sum_i E_i^\dagger E_i$ is an operator-sum decomposition of \mathcal{E}^* .

9.12.18 Trace, transpose, Choi

Let $\mathcal{E}: \mathcal{B}(\mathcal{H}) \rightarrow \mathcal{B}(\mathcal{H}')$, and let $d = \dim \mathcal{H}$ and $d' = \dim \mathcal{H}'$. Show that, for any $(d \times d)$ matrix X and any $(d' \times d')$ matrix Y ,

$$\text{tr}[\mathcal{E}(X)Y] = \text{tr}[\tilde{\mathcal{E}}(X^T \otimes Y)].$$

(For example, if we are interested in the component $\mathcal{E}(X)_{ij} = \langle i | \mathcal{E}(X) | j \rangle$, then we can take $Y = |j\rangle\langle i|$.)

9.12.19 Entanglement witness

Show that, if \mathcal{E} is a positive semi-definite map that is not necessarily completely positive, then its Choi matrix $\tilde{\mathcal{E}}$ is still positive semi-definite on *separable* states.

9.12.20 Almost Kraus decomposition

Show that any linear map $\mathcal{E}: \mathcal{B}(\mathcal{H}) \rightarrow \mathcal{B}(\mathcal{H}')$ can be written as $\rho \mapsto \sum_k E_k \rho F_k^\dagger$. This is very reminiscent of the Kraus decomposition, except that here E_k and F_k are not, in general, the same operator.

Hint: use the singular-value decomposition of the Choi matrix.

9.12.21 Tricks with a maximally mixed state

A maximally mixed state of a bipartite system can be written, using the Schmidt decomposition (from Exercise 5.14.13), as

$$|\Omega\rangle = \frac{1}{\sqrt{d}} \sum_i |i\rangle|i\rangle$$

whence

$$|\Omega\rangle\langle\Omega| = \frac{1}{d} \sum_{i,j} |i\rangle\langle j| \otimes |i\rangle\langle j|$$

Each subsystem is of dimension d , and all the Schmidt coefficients are equal. Here are few useful tricks involving a maximally mixed state.

- If we take the transpose in the Schmidt basis of $|\Omega\rangle$, then

$$\langle \Omega | A \otimes B | \Omega \rangle = \frac{1}{d} \text{tr}(A^T B).$$

- Any pure state $|\psi\rangle = \sum_{i,j} c_{ij} |i\rangle |j\rangle$ of the bipartite system can be written as

$$(C \otimes \mathbf{1}) |\Omega\rangle = (\mathbf{1} \otimes C^T) |\Omega\rangle.$$

This implies that

$$(U \otimes U^*) |\Omega\rangle = |\Omega\rangle$$

(where U^* denotes the matrix given by taking the complex conjugate, entry-wise, of U , i.e. *without* also taking the transpose).

- The swap operation $\text{SWAP} = S: |i\rangle |j\rangle \mapsto |j\rangle |i\rangle$ can be expressed as

$$\begin{aligned} S &= d |\Omega\rangle \langle \Omega|^{T_A} \\ &= d \sum_{i,j} (|i\rangle \langle j|)^T \otimes |i\rangle \langle j| \\ &= d \sum_{i,j} |j\rangle \langle i| \otimes |i\rangle \langle j| \end{aligned}$$

where we write X^{T_A} to mean the partial transpose over \mathcal{A} , i.e. $T \otimes \mathbf{1}$. This implies that

$$\text{tr}[(A \otimes B)S] = \text{tr} AB$$

and that

$$(A \otimes \mathbf{1})S = S(\mathbf{1} \otimes A).$$

Part III

Applications and reality

10 Quantum algorithms

*About quantum interference in disguise: Hadamard, **function evaluation**, Hadamard. Also about the early quantum algorithms and how they deal with querying oracles, searching for a needle in a haystack, and estimating periodicity of certain functions. Finally, about phase estimation, hidden order determination, and Shor's famous algorithm for prime factorisation, via the (inverse) quantum Fourier transform.*

To boil down the theory of classical computers to a single sentence, we can say that they essentially evaluate functions: given n -bits of input, they produce m -bits of output that are uniquely determined by the input. In other words, (very simple) classical computers encode binary functions

$$f: \{0, 1\}^n \rightarrow \{0, 1\}^m$$

and then compute the value of the output for any particular specified n -bit argument. But we can make an even further simplification: a binary function with an m -bit output value is equivalent to m -many binary functions with 1-bit output values (which we call **Boolean functions**). In other words, we might just as well say that the basic task performed by a computer is the evaluation of Boolean functions

$$f: \{0, 1\}^n \rightarrow \{0, 1\}.$$

How can we adapt this to the world of quantum computing?

10.1 Quantum Boolean function evaluation

In quantum computation, *all elementary operations are reversible* (i.e. unitary), so we need to compute Boolean functions in a reversible fashion — we can do so as follows:

$$|x\rangle|y\rangle \mapsto |x\rangle|y \oplus f(x)\rangle.$$

The corresponding circuit diagram (for an input register of $n = 3$ qubits) is shown in Figure 10.1.

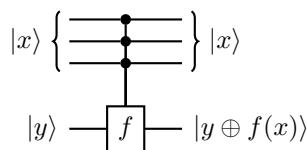


Figure 10.1: Computing some $f: \{0, 1\}^3 \rightarrow \{0, 1\}$ in a quantum manner, where $x \in \{0, 1\}^3$, $y \in \{0, 1\}$, and \oplus denotes XOR, or addition modulo 2.

Here we use two registers: the first one stores the arguments $|x\rangle$ (where $x \in \{0, 1\}^n$ is our binary string input), and the second one the value $f(x)$. More precisely, the value $f(x)$ is added bit-wise to the pre-existing binary value y of the second register. We usually set $y = 0$ to get

$$|x\rangle|0\rangle \mapsto |x\rangle|f(x)\rangle.$$

Quantum Boolean function evaluation is a special case of the generalised x -controlled- U on two registers:

$$\sum_{x \in \{0, 1\}^n} |x\rangle\langle x| \otimes U_x$$

Reading the circuit diagram from top to bottom.

where U_x is either the identity $\mathbf{1}$ (when $f(x) = 0$) or the bit-flip X (when $f(x) = 1$). We can write this very succinctly as

$$\sum_{x \in \{0,1\}^n} |x\rangle\langle x| \otimes X^{f(x)}.$$

Do not confuse the capital X (the Pauli bit-flip operator σ_x) with the small x (a binary string stored in the first register, and the argument of our Boolean function f).

Because of this, we sometimes denote the quantum evaluation of the function f by U_f , which is a gate on the $(n + 1)$ qubits $|x\rangle|y\rangle$.

Let's look at a worked example. Consider the Boolean function $f: \{0, 1\}^2 \rightarrow \{0, 1\}$ given by

$$f(x) = \begin{cases} 1 & \text{if } x = 01; \\ 0 & \text{otherwise} \end{cases}$$

which we might call the **indicator** (or **characteristic**) function for the binary string 01, sometimes denoted χ_{01} . The evaluation $|x\rangle|y\rangle \mapsto |x\rangle|y \oplus f(x)\rangle$ can be tabulated explicitly:

$$\begin{array}{ll} |00\rangle|0\rangle \mapsto |00\rangle|0\rangle & |00\rangle|1\rangle \mapsto |00\rangle|1\rangle \\ |01\rangle|0\rangle \mapsto |01\rangle|1\rangle & |01\rangle|1\rangle \mapsto |01\rangle|0\rangle \\ |10\rangle|0\rangle \mapsto |10\rangle|0\rangle & |10\rangle|1\rangle \mapsto |10\rangle|1\rangle \\ |11\rangle|0\rangle \mapsto |11\rangle|0\rangle & |11\rangle|1\rangle \mapsto |11\rangle|1\rangle \end{array}$$

and then

$$\begin{aligned} \sum_{x \in \{0,1\}^2} |x\rangle\langle x| \otimes X^{f(x)} = & |00\rangle\langle 00| \otimes \mathbf{1} + |01\rangle\langle 01| \otimes X \\ & + |10\rangle\langle 10| \otimes \mathbf{1} + |11\rangle\langle 11| \otimes \mathbf{1}. \end{aligned}$$

Finally, the matrix form looks as follows:

$$\left[\begin{array}{c|cc|cc|cc} 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ \hline 0 & 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 & 0 \\ \hline 0 & 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 \\ \hline 0 & 0 & 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 \end{array} \right]$$

As you can see, this is a diagonal block matrix: a (4×4) matrix with (2×2) matrices as entries. The rows and the columns of the (4×4) matrix are labelled by the binary strings 00, 01, 10, 11, and the (2×2) matrices on the diagonal represent operations applied to the qubit in the second register. Here, all of these matrices on the diagonal are the identity $\mathbf{1}$ except for the $(01, 01)$ entry (i.e. the second one), which is the bit-flip X . This is because $f(01) = 1$ (and so we want to turn the control value $y = 0$ into $y = 1$, which is achieved by applying the bit-flip operator), but $f(x) = 0$ for all other binary strings x (and so we want to leave the control value $y = 0$ as it is).

We always use the **lexicographic order**
 $00 < 01 < 10 < 11$.

10.2 More phase kick-back

What makes quantum evaluation of Boolean functions really interesting — and what truly sets it apart from classical evaluation — is its action on a *superposition of different inputs*. For example,

$$\sum_x |x\rangle|0\rangle \mapsto \sum_x |x\rangle|f(x)\rangle$$

We make two notational simplifications: we usually don't worry about normalisation factors, and we often just write \sum_x to mean $\sum_{x \in \{0,1\}^n}$.

produces $f(x)$ for all x in a *single* run. However, it is more instructive to see the effect of quantum function evaluation when the qubit in the second register is prepared in the state $|-\rangle := \frac{1}{\sqrt{2}}(|0\rangle - |1\rangle) = H|1\rangle$, since then

$$\sum_x |x\rangle |-\rangle \mapsto \sum_x (-1)^{f(x)} |x\rangle |-\rangle$$

(as shown in Figure 10.2). In words, whenever $f(x) = 1$, the bit-flip X is applied to the qubit in the second register.

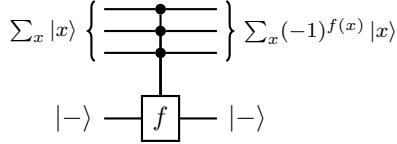


Figure 10.2: Computing some $f: \{0,1\}^3 \rightarrow \{0,1\}$ with the second register in state $|-\rangle$.

The reason for defining the state $|-\rangle$ as we do is that it is the eigenstate of X with eigenvalue -1 , i.e. $X|-\rangle = -|-\rangle$. So, due the phase kick-back, whenever $f(x) = 1$, the phase factor -1 appears in front of the corresponding term $|x\rangle$. As you can see, the second register stays in state $|-\rangle$ all the way through the computation — it is the first register where things happen.

Let us now see how quantum Boolean function evaluation introduces phase shifts in quantum interference experiments, and how such experiments can be viewed as computations.

10.3 Oracles

The computational power of quantum interference was discovered by counting how many times certain Boolean functions have to be evaluated in order to find the answer to a given problem. Imagine a “black box” (sometimes also called an **oracle**) that computes some fixed Boolean function, but whose inner workings are unknown to us. Then imagine that we are in a scenario where we want to learn about some given property of the Boolean function, but we have to “pay” (in energy, time, money, or anything!) for each use (often referred to as a **query**) of the box. In such a setting, the objective is to minimise number of queries to the oracle while finding out as much information as possible about the function that it computes. For this purpose, we ignore everything that happens inside the black box: in our rules of the game, the Boolean function evaluation counts as just *one* computational step.

10.4 Deutsch's algorithm

We start, once more, with the simplest quantum interference circuit

$$|0\rangle \xrightarrow{\quad H \quad} \xrightarrow{\varphi} \xrightarrow{\quad H \quad} \cos \frac{\varphi}{2} |0\rangle - i \sin \frac{\varphi}{2} |1\rangle$$

but let's turn this into a black-box scenario, often known as the **binary observable measurement** problem:

- we are allowed to prepare the input in any state that we like;
- we are allowed to read the output;
- *but* all we know about the value of φ is that it is either 0 or π , and we are *not* allowed to “look inside” the phase gate to see which value it is!

You might recognise this as Exercise 2.14.8, and we will return to this problem again in Section 10.8.

With these rules, can we figure out which value φ has been set to?

Of course we can — we're quantum information scientists!

One way of doing it is to prepare the input in the state $|0\rangle$ and check the output: if $\varphi = 0$ then the output is always $|0\rangle$, and if $\varphi = \pi$ then it is always $|1\rangle$. In other words, a *single run* of the interference experiment is sufficient to determine the difference.

The very first quantum algorithm, proposed by [David Deutsch](#) in 1985, is very much related to this effect, but where the phase setting is determined by the Boolean function evaluation via the phase kick-back.

Scenario. (Global properties of a one-bit function).

We are presented with an oracle that computes some unknown function $f: \{0, 1\} \rightarrow \{0, 1\}$. Note that there are only four possibilities for what f can be: it could be one of two **constant** functions (i.e. those where $f(0) = f(1)$), or one of two **balanced** functions (i.e. those where $f(0) \neq f(1)$).

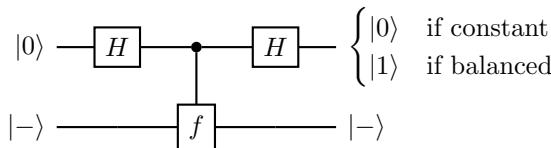
	$f(0)$	$f(1)$
constant	0	0
constant	1	1
balanced	0	1
balanced	1	0

Our task is to determine, using the fewest queries possible, whether the function computed by the oracle is constant or balanced.

Note that we are *not* asked for the particular values $f(0)$ and $f(1)$, but *only whether the two values are the same or different*. Classical intuition tells us that we have to evaluate both $f(0)$ and $f(1)$ and compare them, which involves evaluating f twice. But, in the quantum setting, we can solve this problem with a *single* function evaluation, using the following circuit.

Circuit. (Deutsch's algorithm).

First register: 1 qubit. Second register: 1 qubit.

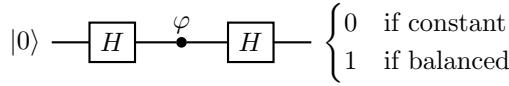


The original version of Deutsch's algorithm provides the correct answer with probability 50%. Here we present a modified/improved version. The more general problem, which deals with unknown functions $f: \{0, 1\}^n \rightarrow \{0, 1\}$ for $n \geq 1$, is known as the [Deutsch–Jozsa problem](#).

During the function evaluation, the second register “kicks back” the phase factor $(-1)^{f(x)}$ in front of $|x\rangle$, but the state of the second register remains unchanged; the first register is modified as follows:

$$\begin{aligned} |0\rangle &\xrightarrow{H} |0\rangle + |1\rangle \\ &\xrightarrow{U_f} (-1)^{f(0)}|0\rangle + (-1)^{f(1)}|1\rangle \\ &= |0\rangle + (-1)^{f(0)\oplus f(1)}|1\rangle \\ &\xrightarrow{H} |f(0) \oplus f(1)\rangle. \end{aligned}$$

The evolution of the first qubit is thus identical to that described by the circuit diagram



where the relative phase is $\varphi = (-1)^{f(0)\oplus f(1)}$. The first qubit ends in state $|0\rangle$ if the function f is constant, and in state $|1\rangle$ if the function is balanced, and the standard measurement distinguishes these two cases with certainty.

But really this is just the binary observable measurement problem in disguise! Indeed, the fact that quantum Boolean function evaluation of a function f is given by

$$|x\rangle|y\rangle \mapsto |x\rangle|y \oplus f(x)\rangle$$

means that the unitary U_f has eigenvalues ± 1 because it satisfies $U_f^2 = \mathbf{1}$, since two consecutive evaluations gives

$$\begin{aligned} |x\rangle|y\rangle &\mapsto |x\rangle|y \oplus f(x)\rangle \\ &\mapsto |x\rangle|y \oplus f(x) \oplus f(x)\rangle \\ &= |x\rangle|y\rangle. \end{aligned}$$

So the fact that $1 = e^0$ and $-1 = e^{i\pi}$ means that U_f acts as a phase gate with phase either 0 or π . We will come back to this link between Deutsch's algorithm and binary observable measurement in Section 10.8.

Deutsch's result laid the foundation for the new field of quantum computation, and was followed by several other quantum algorithms for various problems. They all seem to rest on the same generic sequence:

- a Hadamard transform;
- function evaluation;
- another Hadamard (or Fourier) transform.

As we shall see in a moment, in some cases (such as in Grover's search algorithm) this sequence is repeated several times.

Let us now follow a tour through the three early quantum algorithms, where each one offers a higher-order speed-up when compared to their classical analogues than the last: firstly linear, then quadratic, and finally exponential. After this, we will look at generalising binary observable measurement, the corresponding algorithm analogous to Deutsch's, and how this leads us to arguably the most famous quantum algorithm: Shor's algorithm for prime factorisation.

10.5 The Bernstein–Vazirani algorithm

Scenario. (Hidden inner-product determination).

We are presented with an oracle that computes some unknown function $f: \{0, 1\}^n \rightarrow \{0, 1\}$, but we are promised that f is of the form

$$f(x) = a \cdot x \equiv (a_1 \cdot x_1) \oplus \dots \oplus (a_n \cdot x_n)$$

for some fixed binary string $a = a_1 a_2 \dots a_n \in \{0, 1\}^n$.

Our task is to determine, using the fewest queries possible, the value of the n -bit string a .

It's quite easy to see how to do this classically: if we input the value $x = 00 \dots 010 \dots 0$, where the m -th bit is a 1 and all other bits are 0, then $f(x)$ is simply the m -th bit of a ; after n such calls, we will know every bit value, and thus know a . It is also clear that there cannot exist a better classical algorithm: each call to the oracle teaches us

This is also implemented in the [Quantum Flytrap Virtual Lab](#).

As explained in Section 10.9, the Hadamard transform is a special case of the Fourier transform over the group \mathbb{Z}_2^n .

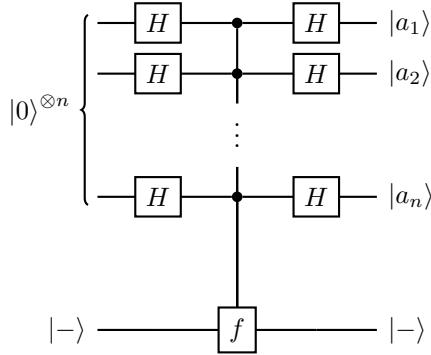
exactly one bit of information, and since we must learn n bits, we must query it n times.

In contrast, by running the circuit below, it is possible to determine the value of a with a *single* call to the oracle!

This algorithm is named for Ethan Bernstein and Umesh Vazirani who proposed it in 1997.

Circuit. (The Bernstein–Vazirani algorithm).

First register: n qubits. Second register: 1 qubit.



A quick note on notation: the “...” in the circuit means “there are more wires here but they are identical (apart from the numbering) to the ones above”. You might also see other notation to denote this, such as

$$|0\rangle^{\otimes n} \xrightarrow{n} H$$

or even simply

$$|0\rangle^{\otimes n} \equiv H$$

Now, stepping through the execution of the circuit (and ignoring the second register, which, as per usual, remains in the state $|-\rangle$ throughout), we obtain

$$\begin{aligned} |0^{\otimes n}\rangle &\xrightarrow{H^{\otimes n}} \left(\frac{1}{\sqrt{2}}\right)^n \sum_{x \in \{0,1\}^n} |x\rangle \\ &\xrightarrow{U_f} \left(\frac{1}{\sqrt{2}}\right)^n \sum_{x \in \{0,1\}^n} (-1)^{a \cdot x} |x\rangle \\ &\xrightarrow{H^{\otimes n}} \left(\frac{1}{\sqrt{2}}\right)^n \sum_{x \in \{0,1\}^n} \left[(-1)^{a \cdot x} \left(\frac{1}{\sqrt{2}}\right)^n \sum_{y \in \{0,1\}^n} (-1)^{y \cdot x} |y\rangle \right] \\ &= \left(\frac{1}{2}\right)^n \sum_{y \in \{0,1\}^n} \left[\sum_{x \in \{0,1\}^n} (-1)^{(a \oplus y) \cdot x} \right] |y\rangle \end{aligned}$$

where we write the second Hadamard transform as

$$|x\rangle \mapsto \left(\frac{1}{\sqrt{2}}\right)^n \sum_{y \in \{0,1\}^n} (-1)^{y \cdot x} |y\rangle.$$

To see that this output is indeed equal to $|a\rangle$, we can use the fact that, for any

Exercise 10.12.5.

$y \in \{0, 1\}^n$,

$$\sum_{x \in \{0,1\}^n} (-1)^{x \cdot y} = \begin{cases} 0 & \text{if } y \neq 0; \\ 2^n & \text{if } y = 0 \end{cases}$$

which, in our case, tells us that

$$\sum_{x \in \{0,1\}^n} (-1)^{(a \oplus y) \cdot x} = \begin{cases} 0 & \text{if } y \neq a; \\ 2^n & \text{if } y = a. \end{cases}$$

In other words, if you take the sum over x , then all the terms always cancel out unless $a \oplus y = 00 \dots 0$, but this happens *if and only if* $y = a$. Then the standard bit-by-bit measurement of the first register gives the value of a and solves the problem with a single call to the oracle.

Alternatively, if you don't immediately see how this sum works for $z \neq a$ (where we write $|z\rangle$ to mean the output), you can first calculate the probability that the output is $z = a$. In this case it is easy to see that the sum is 2^n , and that in the final state $\sum_z \lambda_z |z\rangle$ the term $z = a$ has amplitude 1. Thus, by normalisation, all the other terms must be equal to 0.

10.6 Grover's search algorithm

The next algorithm we will study aims to solve the problem of searching for a specific item in an *unsorted* database. Think about an old-fashioned phone book: the entries are typically sorted alphabetically, by the name of the person that you want to find. However, what if you were in the opposite situation: you had a phone number and wanted to find the corresponding person's name? The phone book is not sorted in that way, and to find the number (and hence name) with, say, 50% probability, you would need to search through, on average, 50% of the entries. Needless to say, in a large phone book this would take a long time.

While this might seem like a rather contrived problem (a computer database *should* always maintain an index on any searchable field), many problems in computer science can be cast in this form, i.e. that of an **unstructured search**.

Scenario. (Unstructured search).

We are presented with an oracle that computes some unknown function $f: \{0, 1\}^n \rightarrow \{0, 1\}$.

Our task is to find, using the fewest queries possible, an input $x \in \{0, 1\}^n$ such that $f(x) = 1$.

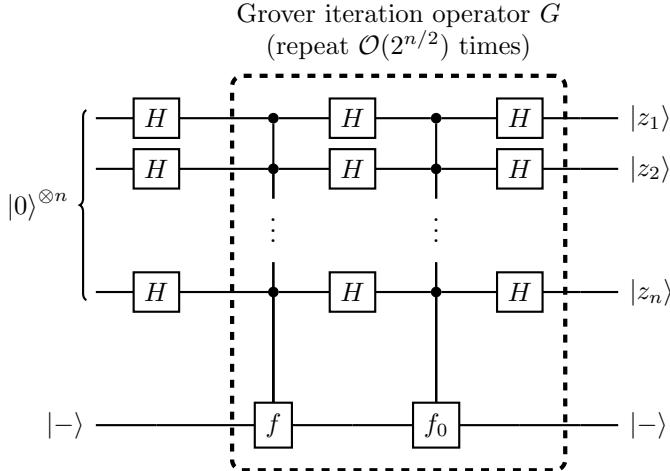
Suppose that we know that, amongst the $N = 2^n$ binary strings, there are $M \ll N$ which are **tagged**, i.e. strings on which f evaluates to 1. Since there is no structure in the database, any classical search requires around N/M steps, i.e. the function f must be evaluated roughly N/M times.

In contrast, there is a quantum search algorithm, implemented by the circuit below, which requires only roughly $\sqrt{N/M}$ steps.

This algorithm is named for [Lov Grover](#), who proposed it in 1996.

Circuit. (Grover's search).

First register: n qubits. Second register: 1 qubit.



where f_0 tags the binary string consisting of n zeros:

$$f_0(x) = \begin{cases} 1 & \text{if } x = 00\ldots 0; \\ 0 & \text{otherwise.} \end{cases}$$

Yet again, we can recognise the typical Hadamard, function evaluation, Hadamard sequence, and yet again we can see that the second register (the bottom qubit, in state $|-\rangle$) plays an auxiliary role: the real action takes place in the first register. However, unlike the previous algorithms, a single call to the oracle does not do very much, and we have to build up the quantum interference in the first register through repeated calls to the oracle (without any intermediate measurements!).

Here, the basic step is the **Grover iteration operator** G , which is the boxed part of the circuit that we repeat over and over:

$$G = (H^{\otimes n} \otimes \mathbf{1})U_{f_0}(H^{\otimes n} \otimes \mathbf{1})U_f.$$

After $\mathcal{O}(2^{n/2})$ applications of G , we measure the first register bit-by-bit and obtain the value of $|z\rangle = |z_1 z_2 \dots z_n\rangle$, which is such that, with “high” probability, $f(z) = 1$. In order to actually see how this algorithm works, and to justify our use of the phrase “with high probability”, we can take a more geometric approach.

First, we define two orthonormal vectors in the Hilbert space describing the first register:

$$|a\rangle = \frac{1}{\sqrt{N-M}} \sum_{x \in f^{-1}(0)} |x\rangle$$

$$|b\rangle = \frac{1}{\sqrt{M}} \sum_{x \in f^{-1}(1)} |x\rangle$$

where $f^{-1}(i) := \{x \in \{0,1\}^n \mid f(x) = i\}$ is the **preimage** of i . Since these vectors are orthonormal, they are, in particular, linearly independent, and so their span is a two-dimensional subspace — this is the subspace in which our search will take place.

Using the fact that $f^{-1}(0)$ and $f^{-1}(1)$ form a **partition** of the space $\{0,1\}^n$, we see that the two-dimensional span of $|a\rangle$ and $|b\rangle$ contains, in particular, the equally-

Recall the big- \mathcal{O} notation introduced in Exercise 1.11.7.

Once again, we shall completely ignore the second register from now on, since all the interesting stuff happens in the first.

A **partition** of a set X is a collection of disjoint subsets $X_1, \dots, X_m \subseteq X$ whose union is all of X , i.e. $X_i \cap X_j = \emptyset$ for all $i \neq j$, and $X_1 \cup \dots \cup X_m = X$.

weighted superposition $|s\rangle = H^{\otimes n}|0^{\otimes n}\rangle$ of all binary strings of length n :

$$\begin{aligned} |s\rangle &= \frac{1}{\sqrt{N}} \sum_{x \in \{0,1\}^n} |x\rangle \\ &= \frac{1}{\sqrt{N}} \sum_{x \in f^{-1}(0)} |x\rangle + \frac{1}{\sqrt{N}} \sum_{x \in f^{-1}(1)} |x\rangle \\ &= \sqrt{\frac{N-M}{N}}|a\rangle + \sqrt{\frac{M}{N}}|b\rangle \\ &= (\cos \theta)|a\rangle + (\sin \theta)|b\rangle \end{aligned}$$

where we use the fact that

$$\sqrt{\frac{N-M}{N}^2} + \sqrt{\frac{M}{N}^2} = 1 = \sin^2 \theta + \cos^2 \theta$$

to parametrise $\sqrt{\frac{N-M}{N}}$ as $\cos \theta$ and $\sqrt{\frac{M}{N}}$ as $\sin \theta$ (with $\theta \approx \sqrt{\frac{M}{N}}$, since $N \gg M$).

The state $|s\rangle$ is our starting input for our sequence of Grover iterations, and we will show that applying G , when restricting to the plane spanned by $|a\rangle$ and $|b\rangle$, amounts to applying a rotation by angle 2θ . Grover's search algorithm can then be understood as a sequence of rotations which take the input state $|s\rangle$ towards the target state $|b\rangle$.

To see this, note that the unitary transformation induced by the oracle

$$f: |x\rangle \mapsto (-1)^{f(x)}|x\rangle$$

can be written as

$$I_a := 2|a\rangle\langle a| - \mathbf{1}$$

which we can interpret as a reflection through the $|a\rangle$ -axis: we see that

$$\begin{aligned} I_a|a\rangle &= 2|a\rangle\langle a||a\rangle - |a\rangle \\ &= 2|a\rangle - |a\rangle \\ &= |a\rangle \\ I_a|b\rangle &= 2|a\rangle\langle a||b\rangle - |b\rangle \\ &= -|b\rangle \end{aligned}$$

and since $-|b\rangle$ is a vector that points in the opposite direction from $|b\rangle$, it must be a reflection; since $-|b\rangle$ is still orthogonal to $|a\rangle$, the reflection must be in the $|a\rangle$ -axis.

Some further algebraic manipulation shows that $I_a = 2|a\rangle\langle a| - \mathbf{1} = \mathbf{1} - 2|b\rangle\langle b|$. Now, in particular, evaluation of f_0 can be written as $2|0\rangle\langle 0| - \mathbf{1}$, and thus thought of as a reflection through the $|0\rangle$ -axis. If we sandwich f_0 in between two Hadamards then we obtain $I_s = 2|s\rangle\langle s| - \mathbf{1}$, which is reflection through the $|s\rangle$ -axis. By definition then, the Grover iteration operator G is the composition

$$G = I_s I_a.$$

Now recall the purely geometric fact that if we have two intersecting lines L_1 and L_2 in two-dimensional Euclidean space, meeting with angle α , then reflecting an object through L_1 and then reflecting the resulting image through L_2 is the same as simply rotating the original object around the point of intersection $L_1 \cap L_2$ by an angle of 2α .

The angle between $|a\rangle$ and $|s\rangle$ is θ , so each time G is applied the vector is rotated (around the origin) by an angle of 2θ towards the $|b\rangle$ -axis. All that remains to do is to just choose the “right” number r of steps such that we end up as close to the $|b\rangle$ -axis

To prove this, it suffices to check that these two transformations agree on the standard basis; since $f^{-1}(0)$ and $f^{-1}(1)$ form a partition, we know that any element (and, in particular, any basis element) is either in the preimage of 0 or of 1; if it is in the former, then I_a acts as the identity; if the latter, then I_a acts as $-\mathbf{1}$.

as possible. The state $|s\rangle$ starts at angle θ to $|a\rangle$, and we should perform our final (and only) measurement when this angle is $\pi/2$, i.e. when $(2r+1)\theta = \pi/2$, which gives

$$r \approx \frac{\pi}{4} \sqrt{\frac{N}{M}}.$$

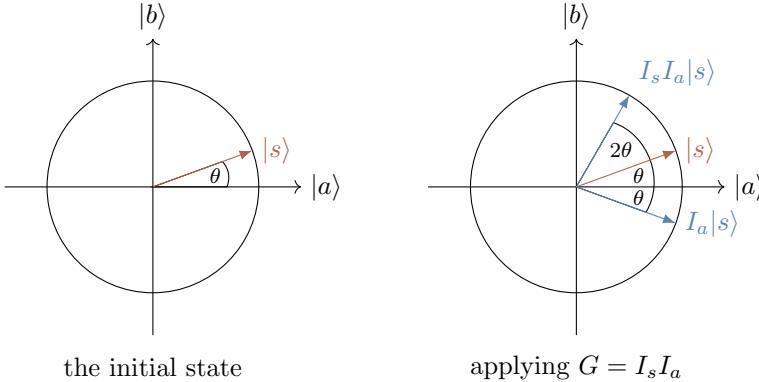


Figure 10.3: Understanding the Grover search algorithm geometrically.

So the quantum algorithm searches an unsorted list of N items in roughly \sqrt{N} steps: it offers a *quadratic* speed-up when compared to classical search, which can be of immense practical importance. For example, cracking some of the more popular modern ciphers, such as [AES](#), essentially requires a search among *many* binary strings (called **keys**). If these can be checked at a rate of, say, one million keys per second, then a classical computer would need over a thousand years to find the correct key, while a quantum computer using Grover's algorithm would find it in less than four minutes.

Recall that $M \ll N$, so $\sqrt{\frac{N}{M}} \approx \sqrt{N}$.

10.7 Simon's algorithm

Here we will see the simplest quantum algorithm that offers an *exponential* speed-up when compared to the best possible classical algorithm.

Scenario. (Hidden binary-addition determination).

We are presented with an oracle that computes some unknown function $f: \{0, 1\}^n \rightarrow \{0, 1\}^n$, but we are promised that f satisfies, for all $x \in \{0, 1\}^n$,

$$f(x) = f(x \oplus s)$$

for some fixed $s \in \{0, 1\}^n$, which we call the **period** of f . (We assume that s is not the string of n zeros, otherwise the problem becomes trivial.)

Our task is to determine, using the fewest queries possible, the value of the n -bit string s .

Note that asking for f to be periodic is equivalent to asking that f be **two-to-one**: for any $y \in \{0, 1\}^n$ such that there exists some $x \in \{0, 1\}^n$ with $f(x) = y$, there exists exactly one other $x' \neq x$ such that $f(x') = y$ as well.

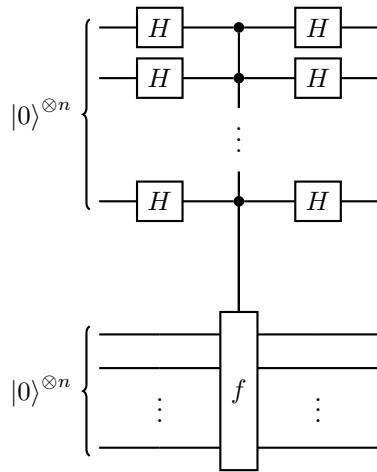
Classically, this problem is exponentially hard. We will not go through a detailed proof of this fact, but the intuition is reasonably simple: since there is no structure in the function f that would help us find its period s , the best we can do is evaluate f on random inputs and hope that we find some distinct x and x' such that $f(x) = f(x')$, and then we know that $s = x \oplus y$. After having made m queries to the oracle, we

have a list of m -many tuples $(x, f(x))$; there are $m(m - 1)/2$ possible pairs which could match within this list, and the probability that a randomly chosen pair match is $1/2^{n-1}$. This means that the probability of there being at least one matching pair within the list of m tuples is less than $m^2/2^n$. This means that the chance of finding a matching pair is negligible if the oracle is queried on fewer than $m = \sqrt{2^n}$ inputs.

The quantum case, on the other hand, gives a result (again, with “high” probability) within a *linear* number of steps. The circuit that solves this problem, shown below, has a familiar Hadamard–function–Hadamard structure, but the second register has now been expanded to n qubits.

This circuit is named for Daniel Simon, who proposed it in 1994.

Circuit. (Simon’s problem).
First register: n qubits. Second register: n qubits.



This time, let’s follow the evolution of *both* registers throughout this circuit. We start off by preparing the equally-weighted superposition of all n -bit strings with the first Hadamard, and then query the oracle:

$$\begin{aligned} |0^{\otimes n}\rangle|0^{\otimes n}\rangle &\xrightarrow{H^{\otimes n} \otimes \mathbf{1}} \frac{1}{\sqrt{2^n}} \sum_x |x\rangle|0^{\otimes n}\rangle \\ &\xrightarrow{U_f} \frac{1}{\sqrt{2^n}} \sum_x |x\rangle|f(x)\rangle. \end{aligned}$$

The second Hadamard transform on the first register then yields the final output state:

$$\frac{1}{2^n} \sum_{x,y} (-1)^{x \cdot y} |y\rangle|f(x)\rangle. \quad (\ddagger)$$

But if we measure the second register *before* applying the second Hadamard transform to the first register, we obtain one of the 2^{n-1} possible values of $f(x)$, each equally likely. Let’s study the implications of this.

As we shall see in a moment, the actual measurement on the second register is not actually necessary.

Suppose that the outcome of the measurement is $f(a)$ for some $a \in \{0, 1\}^n$. Given that both a and $a \oplus s$ are mapped to $f(a)$ by f , the first register then collapses to the state

$$\frac{1}{\sqrt{2}} (|a\rangle + |a \oplus s\rangle).$$

The subsequent Hadamard transform on the first register then gives us the final state

We write s^\perp to mean the set of all $y \in \{0, 1\}^n$ such that $y \cdot s = 0$.

$$\begin{aligned}
 & \frac{1}{\sqrt{2^{n+1}}} \sum_y \left((-1)^{a \cdot y} + (-1)^{(a \oplus s) \cdot y} \right) |y\rangle |f(a)\rangle \\
 &= \frac{1}{\sqrt{2^{n+1}}} \sum_y (-1)^{a \cdot y} \left(1 + (-1)^{s \cdot y} \right) |y\rangle |f(a)\rangle \\
 &= \frac{1}{\sqrt{2^{n-1}}} \sum_{y \in s^\perp} (-1)^{a \cdot y} |y\rangle |f(a)\rangle
 \end{aligned}$$

where we have used the fact that $(a \oplus s) \cdot y = (a \cdot y) \oplus (s \cdot y)$, and that $1 + (-1)^{s \cdot y}$ can have only two values: either 2 (when $s \cdot y = 0$) or 0 (when $s \cdot y = 1$). Now we finally measure the first register: the outcome is selected at random from all possible values of y such that $a \cdot y = 0$, each occurring with probability $1/(2^{n-1})$.

In fact, we did not have to measure the second register at all: it was a mathematical shortcut, simply taken for pedagogical purposes. Instead of collapsing the state to just one term in a superposition, we can express Equation (‡) as

$$\begin{aligned}
 & \frac{1}{2^n} \sum_{y, f(a)} \left((-1)^{a \cdot y} + (-1)^{(a \oplus s) \cdot y} \right) |y\rangle |f(a)\rangle \\
 &= \frac{1}{2^n} \sum_{y, f(a)} (-1)^{a \cdot y} \left(1 + (-1)^{s \cdot y} \right) |y\rangle |f(a)\rangle
 \end{aligned}$$

where the summation over $f(a)$ means summing over all binary strings in the image of f , which is a convenient shorthand for the more complicated formal statement: for all $z \in \{0, 1\}^n$, since our function f is two-to-one, we know that the preimage $f^{-1}(z)$ has either *two* elements or *none*; if it's the latter, then we don't include z in our sum; if it's the former, we *pick one* and call it a (knowing that the other, by the setup, must be $a \oplus s$, which you can see appearing in the first equation above). With this, the final output of the algorithm is

$$\frac{1}{2^{n-1}} \sum_{y \in s^\perp} |y\rangle \sum_{f(a)} (-1)^{a \cdot y} |f(a)\rangle$$

and, again, the measurement outcome is selected at random from all possible values of y such that $s \cdot y = 0$.

We are not quite done yet: we cannot infer s from a *single* output y . However, once we have found $n - 1$ linearly independent strings y_1, y_2, \dots, y_{n-1} , we can solve the $n - 1$ equations

$$\left\{ \begin{array}{l} s \cdot y_1 = 0 \\ s \cdot y_2 = 0 \\ \vdots \\ s \cdot y_{n-1} = 0 \end{array} \right\}$$

to determine a unique value of s . (Note that we only need $n - 1$ values, and not n , because $s = 0$ will always be a solution, but we have explicitly assumed that this is not the case in our statement of the scenario, and so it suffices to narrow down the space of possible solutions to consist of *two* elements, since then we know that we can just take the non-zero one.)

So we run this algorithm repeatedly, each time obtaining another value of y that satisfies $s \cdot y = 0$. Every time we find some new value of y that is linearly independent of all previous ones, we can discard half the potential candidates for s .

Another (complicated sounding) way of expressing this sum is by choice of a **section** of f , i.e. picking *one* element a_z in *each* preimage $f^{-1}(z)$ is equivalent to defining a function $a: \{0, 1\}^n \rightarrow \{0, 1\}^n$ by $a \mapsto a_z$ which satisfies $(a \circ f)(z) = z$, or $a \circ f = \mathbf{1}$.

It is important to note that we are talking about the pure “abstract” *binary strings*, and not the corresponding states in the Hilbert space. Concretely, this means that **linearly independent** here means “no string in the set $\{y_1, \dots, y_n\}$ can be expressed as the bitwise sum of some other strings in this set”. That is, we are working with the $\mathbb{Z}/2\mathbb{Z}$ -vector space of strings, not the \mathbb{C} -vector space of states!

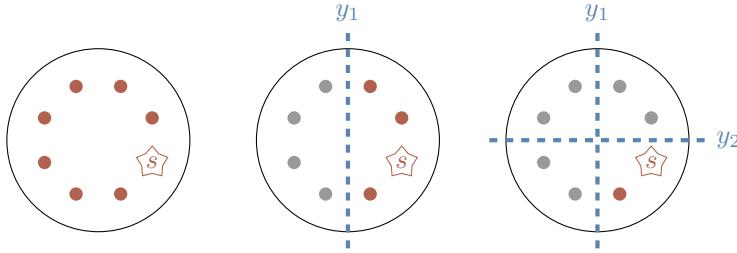


Figure 10.4: Picture all possible binary strings as dots, but with the string s denoted by a star. Every linearly independent y_{k+1} lets us “zoom in” twice as close towards s .

Now, the probability that $(n - 1)$ -many outputs y_1, \dots, y_{n-1} are linearly independent is

$$\left(1 - \frac{1}{2^{n-1}}\right) \left(1 - \frac{1}{2^{n-2}}\right) \cdots \left(1 - \frac{1}{2}\right). \quad (\circledast)$$

To see this, suppose that we have k linearly independent binary strings y_1, \dots, y_k . Then these strings span a subspace with 2^k elements, consisting of all binary strings of the form $\bigoplus_{i=1}^k b_i y_i$, where $b_1, \dots, b_k \in \{0, 1\}$. Now suppose we obtain some y_{k+1} . It will be linearly independent from the y_1, \dots, y_k if and only if it lies *outside* the subspace spanned by the y_1, \dots, y_k , which occurs with probability $1 - (2^k)/(2^n)$.

We can bound Equation (\circledast) from below: the probability of obtaining a linearly independent set $\{y_1, \dots, y_{n-1}\}$ by running the algorithm $n - 1$ times (i.e. not having to discard any values and run again) is

$$\prod_{k=1}^{n-1} \left(1 - \frac{1}{2^k}\right) \geq \left[1 - \left(\frac{1}{2^{n-1}} + \frac{1}{2^{n-2}} + \cdots + \frac{1}{4}\right)\right] \cdot \frac{1}{2} > \frac{1}{4}.$$

We conclude that we can determine s with some constant probability of error after repeating the algorithm $\mathcal{O}(n)$ times. The exponential separation that this algorithm demonstrates between quantum and classical highlights the vast potential of a quantum computer to speed up function evaluation.

10.8 Phase estimation

In Section 10.4, we took the problem of binary observable measurement and at Deutsch’s algorithm, which uses quantum Boolean function evaluation as the controlled- U gate. Now we’re going to generalise the binary observable measurement problem to other controlled- U gates, trying to deduce the value of an unknown phase that can be a lot more general than simply 0 or π . Afterwards, we’ll explain how this is the first step towards Shor’s algorithm for prime factorisation.

Let’s start by rephrasing the binary observable measurement problem in terms of **phase estimation** (or **eigenvalue estimation**). Recall the problem: we are handed some phase gate of unknown phase φ , but we are promised that φ is either 0 or π ; we want to figure out which one it is by running a simple circuit one time. But this is entirely equivalent to having access to an oracle that computes a controlled- U gate, along with having an eigenstate $|u\rangle$ with eigenvalue ± 1 (i.e. $e^{i\varphi}$ for $\varphi \in \{0, \pi\}$), since this acts as a phase gate of phase 0 or π (depending on whether the eigenvalue is $+1$ or -1) on the first register when we plug in $|u\rangle$ to the second register. This is just phase kick-back again! So let’s state the problem this way, and use the solution that we have already described in Section 10.4 (which is really the circuit that we saw all the way back in Section 5.12).

Use the inequality

$$\begin{aligned} (1-x)(1-y) &= 1 - x - y - xy \\ &\geq 1 - (x+y) \end{aligned}$$

which holds for any $0 < x, y < 1$.

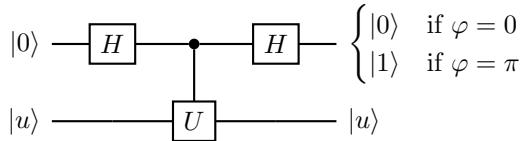
Scenario. (Binary phase estimation).

We are presented with an oracle that computes a controlled- U gate, along with an eigenstate $|u\rangle$ of U with eigenvalue $e^{i\varphi}$. We are promised that φ is either 0 or π .

Our task is to determine, using the fewest queries possible, the value of φ .

Circuit. (Controlled- U interference).

First register: 1 qubit. Second register: 1 qubit.



So we see that the eigenvalues being ± 1 , or, equivalently, φ being 0 or π , is fundamental to this story — this is what we want to generalise, so that we can deal with more values of φ .

Scenario. (Phase estimation).

We are presented with an oracle that computes a controlled- U gate, along with an eigenstate $|u\rangle$ of U with eigenvalue $e^{i\varphi}$. We are promised that φ is of the form

$$\varphi = 2\pi \frac{m}{2^n}$$

for some integers m and n .

Our task is to determine, using the fewest queries possible, the value of $\varphi/2\pi = m/2^n$

We won't be able to find the value of completely arbitrary phases exactly, only those of a certain rational form. However, we will talk about the problem of arbitrary phases in Section 12.9.

Let's work our way up to finding a circuit to solve this problem, starting with the following simple mathematical observation. Since $e^{i\varphi} = e^{i(\varphi+2\pi)}$, we can assume m to be an integer between 0 and $2^n - 1$, which means that it has a binary representation of the form

$$m = \sum_{i=1}^n m_i 2^{n-i}$$

for $m_i \in \{0, 1\}$, and so

$$\varphi = 2\pi \sum_{i=1}^n m_i 2^{-i}$$

This helps us rephrase the problem as “find the values of the m_i for $i = 1, \dots, t$ ”.

Now, rather than tackling the full problem, let's try a much smaller modification of the original binary observable measurement problem: we have the same setup — a phase gate of unknown phase φ — but this time we are promised that φ is either 0 or $\pi/2$ (instead of 0 or π). How can we adapt our original solution to find the phase value?

After some thought, you might see the trick: if we can apply the phase gate twice in a row, then we reduce the problem to the one we have already solved. Indeed, if

$\varphi = 0$ then repeating it twice is the same as simply applying it once; if $\varphi = \pi/2$ then repeating it twice is the same as simply applying a phase gate of phase π .

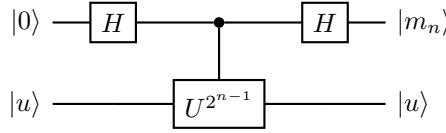
Iterating this idea leads us to the first step of solving the general phase estimation problem, since if $U|u\rangle = e^{i\varphi}|u\rangle$ then $U^{2^{n-1}}|u\rangle = e^{2^{n-1}i\varphi}|u\rangle$. But

$$\begin{aligned} 2^{n-1}\varphi &= 2^n\pi \frac{m}{2^n} \\ &= \pi m \\ &= \pi \sum_{i=1}^n m_i 2^{n-i} \\ &= \pi m_n + \pi \underbrace{\sum_{i=1}^{n-1} m_i 2^{n-i}}_{\text{all divisible by } 2} \end{aligned}$$

and so, by the 2π -periodicity of e^{ix} , we see that

$$U^{2^{n-1}}|u\rangle = e^{m_n\pi i}|u\rangle$$

and now we're happy: $m_n \in \{0, 1\}$, so this is an eigenvalue of $U^{2^{n-1}}$ with phase 0 or π , reducing us to the original binary observable measurement problem that we have already solved by replacing U with $U^{2^{n-1}}$. We have found the n -th bit m_n !



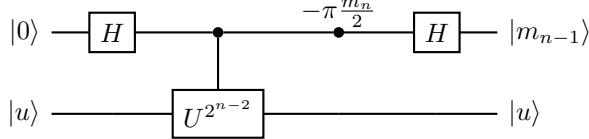
We've dealt with shrinking the distance between the two possible phases (if we have 0 and π/k for some $k \in \mathbb{N}$ then we simply replace U by U^k), so now let's look at the other way of modifying a pair of phases: shifting. We have the same setup — a phase gate of unknown phase φ — but this time we are promised that φ is either $-\pi/2$ or $\pi/2$ (instead of 0 or π). How can we adapt our original solution to find the phase value?

This time we just need to add $\pi/2$ to each of the phases, since then we'd end up back at 0 and π . This means that we simply need to precede (or follow) the phase gate by the $\pi/4$ -phase gate $S = [\begin{smallmatrix} 1 & 0 \\ 0 & i \end{smallmatrix}]$.

We already know how to find the value of m_n , so let's use this to now find the value of m_{n-1} . If we just try the same trick as before then we run into problems: it's true that $U^{2^{n-2}}|u\rangle = e^{2^{n-2}\varphi}|u\rangle$, but

$$2^{n-2}\varphi \equiv \pi m_{n-1} + \pi m_n/2 \pmod{2\pi}$$

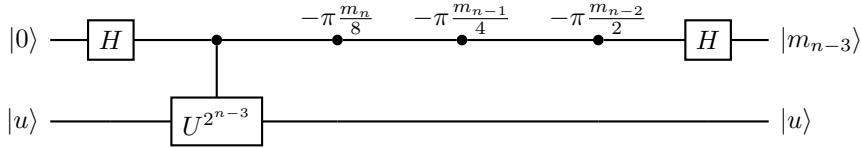
so we have an annoying phase of $\pi m_n/2$ in the way. However, we know that all we have to do now is to compensate for this by adding another phase gate:



Once we've got the value of m_{n-1} from this circuit, then we can simply apply the same idea to find m_{n-2}, m_{n-3} , and so on, until we have all the m_i for $i = 1, \dots, n$. For example, the circuit to determine m_{n-3} (given that we already know m_n, m_{n-1} , and m_{n-2}) is

Any interval $[a, b]$ in \mathbb{R} can be turned into any other interval $[c, d]$ by a single shrink (or scale) and a single shift: multiply a and b by $(d - c)/(b - a)$ to get a' and b' , and then add $c - a'$ to both a' and b' .

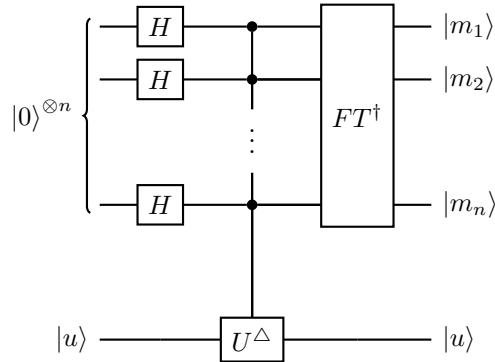
Refer to Section 2.6 for an explanation of this confusing terminology.



The entire process for determining the whole string $m_1 \dots m_n$, solving the scenario, can be written in a form that looks exactly how we expect: Hadamard–phase–Hadamard, modulo some subtle changes, which we will explain.

Circuit. (Phase estimation).

First register: n qubits. Second register: 1 qubit.

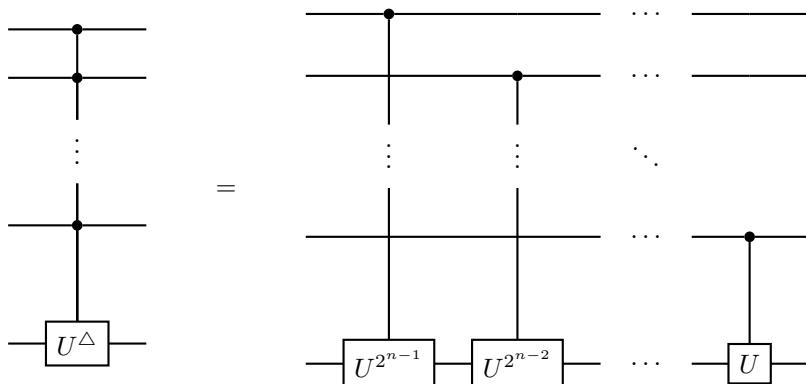


where U^Δ is defined below, and FT^\dagger is defined in Section 10.9.

In this circuit, we write U^Δ to mean the unitary that performs function evaluation as

$$|x\rangle|u\rangle \mapsto e^{i\varphi x}|x\rangle|u\rangle$$

or, in circuit language, the gate that acts on the i -th register as a controlled- $U^{2^{n-i}}$ gate:



It is important to note, however, that we are sweeping something under the rug here. If we can treat U^Δ as an oracle itself, then this does indeed give an efficient solution, but if all we have access to is U then things are definitely not efficient: we are calling it $2^{n-1} + 2^{n-2} + \dots + 2^0 = 2^n - 1$ many times, which is *exponentially bad!*

The only remaining part of the circuit to discuss is this mysterious FT^\dagger gate, but this deserves a section of its own.

10.9 Quantum Fourier transform

In Section 10.8 we constructed a circuit to perform phase estimation of an unknown rational phase $\varphi = 2\pi m/2^n$. The circuit, as per usual, was split into three parts: first some Hadamard gates, then a modified controlled- U gate, and finally some undefined gate FT^\dagger . Following our usual credo, this last gate should look something like a Hadamard, and we shall see that it is indeed related. This gate turns out to be pretty foundational, and turns up all over the place, so it's worth taking some time to talk about it in more detail.

If we look back at what we needed this gate to do, we see that it has to contain these compensating phase shifts that depend on which bit m_i of m we are trying to calculate, e.g. $-\pi(m_n/8 + m_{n-1}/4 + m_{n-2}/2)$ in the case of m_3 . We draw this as a circuit in Figure 10.5.

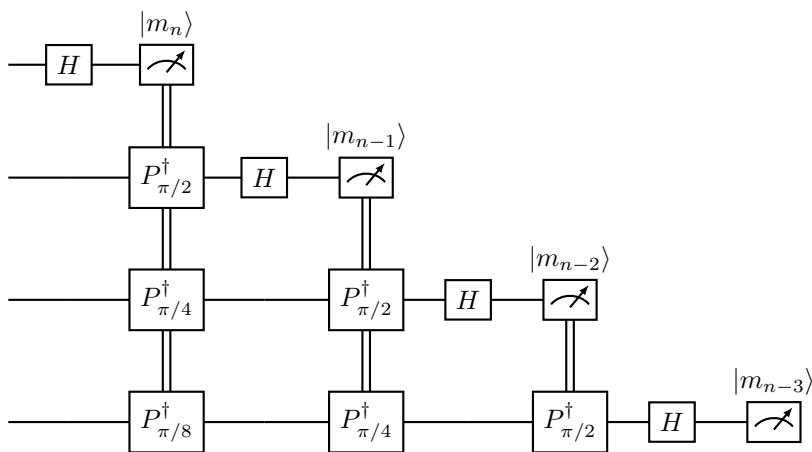


Figure 10.5: The gate FT^\dagger followed by measurement, acting on four qubits. The double vertical lines linking gates indicate that the result of the classical measurement is used to control the application (for example, we want the first phase gate on each register to be multiplied by m_n).

Hopefully it's clear how to generalise this definition of the gate to an arbitrary number of qubits, but to give a formal definition it's easier (and also useful) to give a symbolic definition.

The **quantum Fourier transform (QFT)** acting on n qubits is defined by the unitary

$$U_{\text{FT}} = \frac{1}{\sqrt{N}} \sum_{x,y=0}^{N-1} e^{2\pi i xy/N} |y\rangle\langle x|$$

where $N = 2^n$.

Note that this is the definition for the gate that would be denoted by FT , whereas what we use in the phase estimation circuit is FT^\dagger . In other words, we are really using the **inverse** quantum Fourier transform.

Here we switch from thinking of binary strings of length n to thinking of integers from 0 to $2^n - 1$, as described in Section 5.4. It's good to be comfortable switching between the two!

There are some confusing conventions when it comes to whether or not to use the word "inverse" here, but we generally decide to do so.

Checking that this matrix is actually unitary is slightly involved, so we'll only sketch how it's done here. We can see that

$$U_{\text{FT}} U_{\text{FT}}^\dagger = \frac{1}{N} \sum_{x,y,z=0}^{N-1} e^{2\pi i x(y-z)/N} |y\rangle\langle z|$$

since $|x\rangle\langle y| = \delta_{xy}$, and then we can separately consider the cases $y = z$ and $y \neq z$; in the latter case, we'll have to sum a geometric progression, which gives

$$\begin{aligned} \frac{1}{N} \sum_{x=0}^{N-1} e^{2\pi i x(y-z)/N} &= \frac{1 - e^{2\pi i (y-z)/N}}{1 - e^{2\pi i (y-z)/N}} \\ &= 0. \end{aligned}$$

So in what way does this relate to the Hadamard gate that we already know and love? Looking at the $n = 1$ case of the definition of U_{FT} we get

$$U_{\text{FT}} = \frac{1}{\sqrt{2}} \begin{bmatrix} 1 & 1 \\ 1 & -1 \end{bmatrix}$$

which is exactly the usual Hadamard gate H on 1 qubit. However, if we now look at $n = 2$ and compare U_{FT} with $H \otimes H$ (which we calculated in Exercise 5.14.12), then we see that they differ:

$$\begin{aligned} U_{\text{FT}} &= \frac{1}{2} \begin{bmatrix} 1 & 1 & 1 & 1 \\ 1 & i & -1 & -i \\ 1 & -1 & 1 & -1 \\ 1 & -i & -1 & i \end{bmatrix} \\ H \otimes H &= \frac{1}{2} \begin{bmatrix} 1 & 1 & 1 & 1 \\ 1 & -1 & 1 & -1 \\ 1 & 1 & -1 & -1 \\ 1 & -1 & -1 & 1 \end{bmatrix} \end{aligned}$$

In fact, we can give a general expression for U_{FT} on n qubits: it is the $(2^n \times 2^n)$ matrix

$$U_{\text{FT}} = \frac{1}{\sqrt{2^n}} \begin{bmatrix} 1 & 1 & 1 & 1 & \dots & 1 \\ 1 & \omega & \omega^2 & \omega^3 & \dots & \omega^{2^n-1} \\ 1 & \omega^2 & \omega^4 & \omega^6 & \dots & \omega^{2(2^n-1)} \\ 1 & \omega^3 & \omega^6 & \omega^9 & \dots & \omega^{3(2^n-1)} \\ \vdots & \vdots & \vdots & \vdots & \ddots & \vdots \\ 1 & \omega^{2^n-1} & \omega^{2(2^n-1)} & \omega^{3(2^n-1)} & \dots & \omega^{(2^n-1)(2^n-1)} \end{bmatrix}$$

where ω is a primitive 2^n -th root of unity (that is, $\omega^{2^n} = 1$, but $\omega^m \neq 1$ for any $m < 2^n$).

It turns out the quantum Fourier transform and the Hadamard are both examples of a more general construction known as the [Fourier transform on finite groups](#): applied to the cyclic group $\mathbb{Z}/2^n\mathbb{Z}$ we get the QFT; applied to the n -fold product $(\mathbb{Z}/2\mathbb{Z})^n$ we get the Hadamard. However, we won't need this level of formalism going forward, we just need to remember that we have yet one more useful gate at our disposal!

Fourier theory and group representations.

This section is not yet finished.

The way that we wrote the QFT in Figure 10.5 is not the conventional way in which it normally appears, since we're assuming that we measure the result at the

end. In general, we might not want to do this, and instead replace the dependence of the corrective phase gates on the classical measurements by controlled qubits, as shown in Figure 10.6. We sometimes refer to this as the **coherent** circuit for the inverse quantum Fourier transform.

One other note about conventions is the difference in ordering of bits between the circuit representation above and the unitary operator definition given before: in the circuit, the least significant bit appears in the *first* register as opposed to the last, with increasing significance of bits as we go *up* registers (i.e. down the page). We could of course introduce a lot of SWAP gates to make these conventions agree, but this is typically unnecessary if we instead just keep track of which qubits are which and remember what we want to do with each one next.

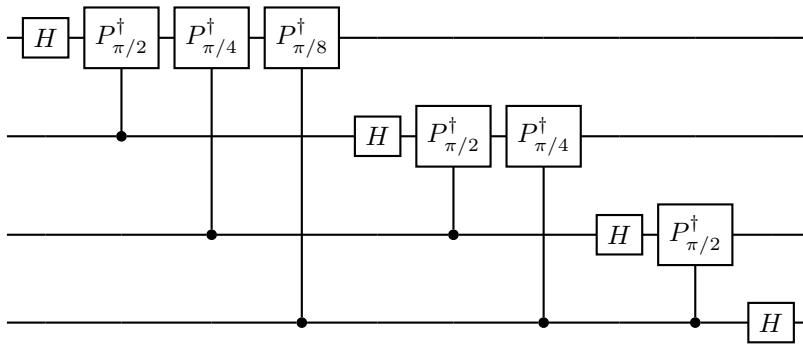


Figure 10.6: The gate FT^\dagger without measurement, acting on four qubits. This no longer depends on any classical measurements, but instead uses controlled-phase gates.

10.10 Hidden-order determination

In the same way that Deutsch's algorithm (10.4) dealt with binary observable measurement for a unitary implementing some function evaluation, the hidden-order determination algorithm that we are now going to study deals with phase estimation for a unitary implementing some function evaluation. This turns out to be an important step along the path towards Shor's algorithm

Deutsch's algorithm is to binary observable measurement as hidden-order determination is to phase estimation.

Scenario. (Hidden-order determination).

We are presented with an oracle that computes the function on n qubits

$$U_a|z\rangle = |az \bmod N\rangle$$

for some fixed (known) integer $N = 2^n$, along with a fixed (known) integer a that is coprime to N .

Our task is to determine, using the fewest queries possible, the order r of a modulo N , i.e. the smallest positive integer r such that $a^r \equiv 1 \pmod{N}$.

Let's be clear and spell out what notational shortcuts we're making here. Every integer $0 \leq z < 2^n$ corresponds to a basis state $|z\rangle$ of the space of n qubits, given by writing z in binary form. So when we write $|az \bmod N\rangle$, this means that we take the product of the integers a and z , take this integer modulo N , and then consider the basis state given by writing this result in binary form.

The hypothesis that N is a power of 2 is actually unnecessary, as we will later explain, but we assume it for now since for the sake of ease.

For example, if $n = a = 3$, then U_3 applied to $|4\rangle = |1\rangle|0\rangle|0\rangle$ is $|12 \bmod 8\rangle = |4\rangle = |1\rangle|0\rangle|0\rangle$, and applied to $|5\rangle = |1\rangle|0\rangle|1\rangle$ is $|15 \bmod 8\rangle = |7\rangle = |1\rangle|1\rangle|1\rangle$.

First of all, we can write down some eigenstates of U_a parametrised by an integer $s \in \mathbb{Z}$

$$|u_s\rangle = \frac{1}{\sqrt{r}} \sum_{k=0}^{r-1} e^{-2\pi i (sk/r)} |a^k \bmod N\rangle.$$

Since $2\pi i(r+1)k/r \equiv 2\pi i(k/r) \bmod 2\pi i$, we see that this gives us r eigenstates, parametrised by $s = 0, \dots, r-1$. Now let's show that these are indeed eigenstates. By the definition of U_a ,

$$\begin{aligned} U_a |u_s\rangle &= \frac{1}{\sqrt{r}} \sum_{k=0}^{r-1} e^{-2\pi i (sk/r)} |y^{k+1} \bmod N\rangle \\ &= \frac{1}{\sqrt{r}} \sum_{j=1}^r e^{2\pi i (s/r)} e^{-2\pi i (sj/r)} |y^j \bmod N\rangle \end{aligned}$$

where we simply changed the index of the sum to $j = k + 1$. But r is the order of a modulo N , so we the $j = r$ term can be written as $j = 0$, giving

$$U_a |u_s\rangle = e^{2\pi i (s/r)} |u_s\rangle$$

which shows that $|u_s\rangle$ is an eigenstate with eigenvalue $e^{2\pi i (s/r)}$.

Of course, we cannot actually prepare any of these eigenstates $|u_s\rangle$ because they require knowledge of the natural number r that we are trying to find! But at the very least this now looks a bit like a problem that we have seen before: if we *could* prepare some particular $|u_s\rangle$ then the phase estimation algorithm (from Section 10.8) would allow us to calculate s/r , and thus r . So let's try one of our always-useful tricks: superposition.

Although we cannot prepare any $|u_s\rangle$ individually, note that

$$\begin{aligned} \frac{1}{\sqrt{r}} \sum_{s=0}^{r-1} |u_s\rangle &= \frac{1}{r} \sum_{s=0}^{r-1} \sum_{k=0}^{r-1} e^{-2\pi i (sk/r)} |a^k \bmod N\rangle \\ &= \frac{1}{r} \sum_{k=0}^{r-1} \left(\sum_{s=0}^{r-1} e^{-2\pi i (sk/r)} \right) |a^k \bmod N\rangle \\ &= \underbrace{\frac{1}{r} \sum_{s=0}^{r-1} |1\rangle}_{k=0} + \frac{1}{r} \sum_{k=1}^{r-1} \left(\sum_{s=0}^{r-1} e^{-2\pi i (sk/r)} \right) |a^k \bmod N\rangle \\ &= |1\rangle \end{aligned}$$

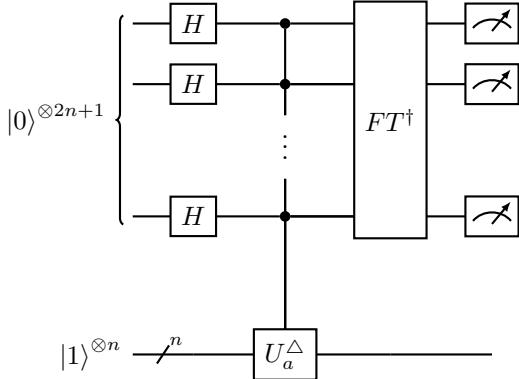
Recall that, if $\omega \neq 1$ is an n -th root of unity, then $\sum_{i=0}^{n-1} \omega^i = 0$.

and so we should get something interesting by just plugging in $|1\rangle$.

The actual circuit is practically identical to the phase estimation circuit of Section 10.8, but where U is replaced by U_a , and $|u\rangle$ is replaced by $|1\rangle$.

Named for [Peter Shor](#), who proposed it as part of his factoring algorithm (see Section 10.11) in 1994.

Circuit. (Shor's order-finding algorithm).
First register: $2n + 1$ qubits. Second register: n qubits.



As always, let's step through the circuit. Immediately after the first Hadamard we are in the state

$$\left(\frac{1}{\sqrt{N}} \sum_{x=0}^{N-1} |x\rangle \right) |1\rangle.$$

Next, just after the function evaluation U_a , we are in the state

$$\frac{1}{\sqrt{r}} \sum_{s=0}^{r-1} |\Phi_s\rangle |u_s\rangle$$

where

$$|\Phi_s\rangle = \frac{1}{\sqrt{N}} \sum_{n=0}^{N-1} e^{i\phi_s x} |x\rangle$$

and $\phi_s = 2\pi(s/r)$, so that $e^{i\phi_s x}$ is the eigenvalue of $|u_s\rangle$ for U_a .

To calculate the effect of the inverse quantum Fourier transform, we will make the same pedagogical shortcut as we did when studying Simon's algorithm (Section 10.7) and pretend that we make a measurement in the basis $|u_s\rangle$ in the auxiliary system *before* applying the inverse QFT. We will measure one of the r possible values of s , each with equal probability $1/r$, and doing so will leave the main register in the corresponding state $|\Phi_s\rangle$. But this is exactly the picture of what happened after applying U^Δ in the phase estimation circuit! This means that we already know what the inverse QFT will do: it will give us the value of $2^{2n+1}(s/r)$. Since this will happen irrelevant of which value of s we measure, we need not actually perform the measurement.

So running through the entire circuit and then measuring the main register will give us the value of $2^{2n+1}(s/r)$, which is *almost* enough to recover the value of r , but not quite: since we didn't measure the auxiliary register, we don't know which value of s we obtained! However, now a purely classical (and efficient) algorithm comes to our rescue: the **continued fractions algorithm**. This will give us the values of s and r provided that a certain inequality is satisfied, which is always the case if our main register has $2n + 1$ qubits. We will not delve into these details of the classical post-processing of the measurements since they are no longer quantum; consider this yet another chance for the interested reader to either work out the details themselves or to find the answers elsewhere through some research of their own.

There is a deep link between this algorithm and Simon's algorithm. As we have mentioned, hidden-order determination forms a key part of Shor's algorithm; Simon's algorithm and Shor's algorithm (and Deutsch's algorithm, in fact) are all closely related: they are all examples of [hidden subgroup problems](#).

It is very fortunate that we do not need to perform this measurement, since in general we will not know how to implement it: recall that to construct $|u_s\rangle$ we would need to already know the value of the order r that we are trying to find.

The explicit inequality in the continued fractions algorithm actually tells us that $2n + 1$ is the smallest possible size for the main register to ensure exactness with complete certainty.

Finally, the assumption at the very start that N be of the form 2^n is not actually necessary (just as the assumption that the phase in the phase estimation problem be of the form $2\pi m/2^n$ is not actually necessary). Suppose that N isn't of this form, and let n be the smallest integer such that $N < 2^n$. Then we want to extend U_a to act on n qubits, and we need to be careful how we do so. For example, we cannot simply take the same function $U_a|z\rangle = |az \bmod N\rangle$, since then $U_a|0\rangle = |0\rangle = U_a|N\rangle$ but $|0\rangle$ and $|N\rangle$ are orthogonal (remember, we always assume our bases to be orthonormal!), so this U_a cannot be unitary because it sends orthogonal states to non-orthogonal states. The correct modification to make for our purposes is to define

$$U_a|z\rangle = \begin{cases} |az \bmod N\rangle & 0 \leq z < N \\ |z\rangle & R \leq z < 2^n \end{cases}$$

which is indeed unitary over the whole space of n qubits.

10.11 Shor's algorithm

Given its importance in the field of (post-)quantum cryptography (and thus in the real world), as well as its fame, it would be remiss of us to not dedicate at least one section in this book to Shor's algorithm. However, as you will see, the actual quantum part of the algorithm is exactly the order-finding circuit from Section 10.10; the rest is an application of classical number theory to reduce the problem of prime factorisation to the problem of hidden-order determination. Because of this, we will not spend too much time on the details — this is not a book on classical number theory! — but we hope that this section at least highlights the important fact which is that *other areas of mathematics and physics have much to offer for the aspiring quantum information scientist*. Of course, it is infeasible to try to learn *everything* in mathematics, but this is also exactly the point: it's good to be aware that there is a lot of it out there, and that we all have a lot to learn from each others' specialities.

The purpose of Shor's algorithm is to find the prime factorisation of a composite integer. This is a classically difficult problem, and hence forms the basis of some very well-known public-key cryptography schemes, such as RSA (see Exercise 10.12.1), but Shor's algorithm offers a distinct speed-up. More precisely, to factor an integer R classically with the general number field sieve method is a **sub-exponential** problem (i.e. somewhere between polynomial in R and exponential in R), whereas Shor's algorithm is **polylogarithmic** (i.e. polynomial in $\log R$). This means that Shor's algorithm witnesses integer factorisation as a bounded-error quantum polynomial time problem: it lives in BQP (recall Section 1.9).

Since any integer has a unique (up to ordering) prime factorisation consisting of finitely many prime numbers, it suffices to find a single prime factor, since then we can simply divide our original number by this and run the algorithm over and over until it terminates, recording the prime factors that we find along the way. But still, turning the problem of “find a prime factor” into a problem that we have already solved (namely hidden-order determination) requires some work, so let's get started.

Recall that the **greatest common divisor** $\gcd(m, n)$ (or **highest common factor** $\text{hcf}(m, n)$) of a pair of integers (m, n) is the largest integer that divides both of them. This can be efficiently computed by a (very old) classical algorithm known as **Euclid's algorithm**, so we can make use of it in freely. In fact, let us now list the things that we can do efficiently with classical algorithms:

- compute the hcf of two integers (Euclid's algorithm)
- determine if an integer is prime (the **Miller-Rabin primality test**)
- determine if an integer is even (check if the last digit is a 0 in its binary expansion)
- determine if an integer R can be written as $R = a^b$ for some integers $a \geq 1$ and $b \geq 2$ (Exercise 10.12.6).

There are a few algorithms known by the name of “Shor's algorithm”, but they are all somewhat related.

The fact that prime factorisation is essentially unique is so important that it earns the name of the **fundamental theorem of arithmetic**.

One more small note of terminology: throughout this section, we say **factor** to mean non-trivial factor, i.e. the factors of R are the natural numbers that divide R apart from 1 and R .

Lemma. Let x and R be natural numbers such that

- R is not prime;
- $1 < x < R - 1$;
- $x^2 \equiv 1 \pmod{R}$.

Then both $\text{hcf}(x - 1, R)$ and $\text{hcf}(x + 1, R)$ are factors of R .

Note that $1 < x < R - 1$ is equivalent to saying that $x \not\equiv \pm 1 \pmod{R}$. In other words, we assume x to be a non-trivial square root of 1 modulo R .

Proof. The hypothesis tells us that

$$x^2 - 1 = kR$$

for some integer k , and so factoring $x^2 - 1$ tells us that

$$R \mid (x + 1)(x - 1)$$

(where we write $a \mid b$ to mean that a divides b).

Now, since $x < R - 1$, we know that $x + 1 < R$ and so R cannot divide $x + 1$ because it is simply too big. This means, in particular, that $\text{hcf}(x + 1, R) \neq R$. But now note that if $\text{hcf}(x - 1, R) = 1$ then, since $R \mid (x + 1)(x - 1)$, it must be the case that $R \mid x + 1$, and we have just said that this cannot happen. The same argument applies if we swap $x + 1$ and $x - 1$, so we have proven that both $\text{hcf}(x - 1, R)$ and $\text{hcf}(x + 1, R)$ are factors of R . \square

The essence of the above proof is much simpler than the length might suggest: we are sort of just saying that R cannot divide either $x + 1$ or $x - 1$ alone, and so must be “split up” into two parts, with one inside $x + 1$ and the other inside $x - 1$.

Or we could appeal directly to the fundamental theorem of arithmetic. Write $R = p_1^{a_1} \dots p_n^{a_n}$, and now consider each prime factor individually. Since $R \mid (x + 1)(x - 1)$, we know that $p_1 \mid (x + 1)(x - 1)$. A very useful fact about prime numbers is that, if p divides a product ab , then either p divides a or p divides b . So here we see that p_1 must divide either $x + 1$ or $x - 1$. We can continue like this for all the prime factors p_i of R , but note that it cannot be the case that all the p_i divide only $x + 1$ and not $x - 1$, since this would then force $p_1^{a_1} \dots p_n^{a_n} = R$ to divide $x + 1$, which we have already said cannot happen (and similarly for all the p_i dividing only $x - 1$). This means that both $\text{hcf}(x - 1, R)$ and $\text{hcf}(x + 1, R)$ are non-trivial.

Now let's state another useful lemma (which we will not prove).

This “very useful fact” is secretly just the fact that prime numbers are **prime elements** in the ring of integers. The usual definition of “prime number” that we are used to (“only divisible by 1 and itself”) actually says that they are **irreducible elements**. The link between these two concepts requires some knowledge of ring theory.

Lemma. Let R be an odd natural number with $m \geq 2$ distinct prime factors. If y is chosen uniformly at random from the set of natural numbers that are coprime to and smaller than R , then the probability that the order r of y modulo R is even and satisfies

$$y^{r/2} \not\equiv -1 \pmod{R}$$

is at least

$$1 - \frac{1}{2^m - 1}.$$

How is this lemma useful to us? Well, it says that if we randomly pick some $1 < y < R$ satisfying the hypotheses, then with non-zero probability (that increases as R has more and more prime factors) it will be such that we can apply the previous lemma to $x = y^{r/2}$, *almost*. The one problem is that $x > y$, and so it might also be the case that $x > R$, and the previous lemma needed the assumption that $x < R - 1$. But we can fix this!

Lemma. Let y and R be natural numbers such that

- R is not prime;
- the order r of y modulo R is even;
- $y^{r/2} > R$;

Then either $x + 1$ is divisible by R , or both $\text{hcf}(y^{r/2} - 1, R)$ and $\text{hcf}(y^{r/2} + 1, R)$ are factors of R .

Proof. First of all, recall that the order r is the smallest natural number such that

$$y^r \equiv 1 \pmod{R}.$$

Since r is even, $y^{r/2}$ is well defined and we can factor

$$kR = y^r - 1 = (y^{r/2} + 1)(y^{r/2} - 1)$$

as before. Now we have to do something different, because it is no longer necessarily the case that $y^{r/2} + 1$ is smaller than R . However, we can still show that R does not divide the $y^{r/2} - 1$ term, since if it did then we would have that

$$y^{r/2} \equiv 1 \pmod{R}$$

which contradicts the fact that r is the *smallest* natural number such that this holds. So either R divides the $y^{r/2} + 1$ term, or it doesn't; in the latter case, we can then apply exactly the same argument as before to show that both $\text{hcf}(y^{r/2} - 1, R)$ and $\text{hcf}(y^{r/2} + 1, R)$ are factors of R . \square

The fact that this lemma requires R to be odd and also have at least two distinct prime factors means that we can only apply it if we have checked that both of these things are true. In other words, we need to know that R is not even, and also that R is not of the form p^b for some prime p . But as we've already mentioned, both of these properties can be checked by an efficient algorithm by a classical computer.

Now we're ready to state Shor's algorithm.

Shor's algorithm. (*Factoring the natural number R .*)

1. Check that R is not prime.

If so, stop and return the factor R .

2. Check if R is even.

If so, stop and return the factor 2.

3. Check if $R = a^b$ for some integers $a \geq 1$ and $b \geq 2$.

If so, stop and return the factor a .

4. Uniformly at random pick an integer $1 < y < R$ and evaluate $\text{hcf}(y, R)$; check if $\text{hcf}(y, R) > 1$.

If so, stop and return the factor $\text{hcf}(y, R)$.

5. Compute the order r of y modulo R via hidden-order determination (from Section 10.10).

6. Check if r is odd.

If so, go back to step 4.

7. Check if $y^{r/2} \equiv -1 \pmod{R}$.

If so, go back to step 4.

8. Compute $\text{hcf}(y^{r/2} - 1, R)$ and $\text{hcf}(y^{r/2} + 1, R)$.

Stop and return these as factors.

Picking it apart, we see that steps 1 to 3 are simply checking easy cases; steps 4, 6, and 7 are checking that the necessary hypotheses are satisfied in order for step 8 to calculate factors of R . The only place that we use anything quantum in step 5, where we have to do hidden-order determination. The fact that steps 6 and 7 won't cause us to get stuck in an endless loop is justified by the fact that they will both be passed over with probability $1 - 1/(2^{m-1})$, as we mentioned above.

Even better, thanks to [a result by Ekerå](#), one can be rather sure that this quantum subroutine will only need to be run a single time.

10.12 Remarks and exercises

10.12.1 RSA

This section is not yet finished.

10.12.2 More complexity classes

This section is not yet finished.

10.12.3 Implementing reflections

This section is not yet finished.

10.12.4 Grover's optimality

This section is not yet finished.

10.12.5 Picking out a single state

Prove that, for any $y \in \{0, 1\}^n$,

$$\sum_{x \in \{0,1\}^n} (-1)^{x \cdot y} = \begin{cases} 0 & \text{if } y \neq 0; \\ 2^n & \text{if } y = 0. \end{cases}$$

10.12.6 Writing an integer as a power

1. Show that $R = 21$ cannot be written in the form a^b for integers $a \geq 1$ and $b \geq 2$.
2. Generalise this to a method that could work in $\mathcal{O}(L^3)$ for any value of R that is L bits long.

Hint: since R is L bits long, $R < 2^L$, and so $b < L$.

11 Quantum cryptography

About ...

This section is not yet finished.

12 Approximation

About quantifying precision in implementations of quantum circuits using the notion of **metrics** — more specifically, the **trace distance**. Also about the practical feasibility of universal sets of gates and correctly distinguishing non-orthogonal states described by density operators.

We have talked a lot about preparing specific quantum states and constructing specific unitary operations, but the space of states of any quantum system is a continuous space, and the set of unitary transformations is also continuous. It is entirely unrealistic to imagine that in the actual world we will be able to prepare, for example, a qubit *precisely* in the state $|0\rangle$, or to perform a unitary transformation that is *exactly* equal to the controlled-not gate. *We never have infinite precision in our manipulations of the physical world.* The good news is that, for all practical purposes, infinite precision is not actually necessary, and we can achieve most of our goals by preparing quantum states and performing quantum operations that are “close enough” to the desired ones. But what is “close enough”, and how do we quantify it?

12.1 Metrics

To begin with, let us work with pure states, and save the problem of dealing with mixed states for a later section. We will start with the second question: how do we quantify this notion of “close enough”? The central concept is one with which you are probably already somewhat familiar (we mentioned it in Sections 0.3 and 0.5), namely that of a **metric**, or **distance**.

Given a set X , a **metric** (or **distance**) on X is a function $d: X \times X \rightarrow \mathbb{R}_{\geq 0}$ such that

- **Identity of indiscernibles:** $d(a, b) = 0$ if and only if $a = b$
- **Symmetry:** $d(a, b) = d(b, a)$ for all $a, b \in X$
- **Triangle inequality:** $d(a, c) \leq d(a, b) + d(b, c)$ for all $a, b, c \in X$.

Generalisations of metrics.

There are four conditions governing metrics (identity of indiscernibles is an “if and only if” statement, so we can separate it into two “if” statements). As is usually the case in mathematics, it is interesting to ask what happens if we drop one or more of these.

- If we drop $d(a, b) = 0 \implies a = b$ then we get **pseudometrics**.
- If we drop $a = b \implies d(a, b) = 0$ then we get **metametrics**, or **partial metrics**.
- If we drop $d(a, b) = d(b, a)$ then we get **quasimetrics**. These arise “in real life”, if you think about travelling around a city that has lots of one-way streets, or travelling up or down a big hill.
- If we drop $d(a, c) \leq d(a, b) + d(b, c)$ then we get **semimetrics** (though be careful here: lots of authors use “semimetric” to mean almost any one of these generalisations, and the terminology is very non-consistent!).

We can also consider the case of **extended metrics**, where the distance function is allowed to take the value ∞ . For many category theorists, “the”

notion of metric space is that of an *extended pseudoquasimetric*.

The most common norm is the **Euclidean distance**, that is, distance between two points in Euclidean space. Given points $A = (a_1, a_2, \dots, a_n)$ and $B = (b_1, b_2, \dots, b_n)$ in \mathbb{R}^n , their Euclidean distance is

$$\sqrt{|b_1 - a_1|^2 + |b_2 - a_2|^2 + \dots + |b_n - a_n|^2}.$$

But we already know that Euclidean space \mathbb{R}^n is more than just a set: it is a vector space. This means that we don't just have a metric space (i.e. a set with a metric), but instead a **normed vector space**, where the **norm** $\|\cdot\|$ of a vector is defined to be the distance of that vector from the origin: $\|a\| := d(a, 0)$.

It turns out that this norm (and thus this metric) actually arises from a more fundamental structure, namely that of the **inner product**. Returning to the bra-ket notation, we recall that the norm of any vector $|a\rangle$ is exactly $\|a\| = \sqrt{\langle a|a\rangle}$, and thus the distance between any two vectors $|a\rangle, |b\rangle$ is exactly $d(|a\rangle, |b\rangle) = \||b\rangle - |a\rangle\|$ (though for simplicity we sometimes write this as $\|b - a\|$ instead, or even $\|a - b\|$, since this is equal). This norm is also called the **2-norm**, or the **ℓ^2 -norm** (for reasons that we will come back to in Section 12.11.2), and is defined for any finite-dimensional Hilbert space \mathbb{C}^n using the fact that $\mathbb{C} \cong \mathbb{R}^2$, so that $\|(x, y)\| := \|(x, y)\| = \sqrt{x^2 + y^2}$.

Before moving on to talk about state vectors, let us first discuss one other metric space which shows up in information theory (both classical and quantum). The space of binary strings (of some fixed length n) admits a metric known as the **Hamming distance**. This is defined quite simply as “the number of positions at which the corresponding bits are different”. For example,

$$d(0101101011, 1101110111) = 4$$

since these two strings differ in four bits:

$$\begin{array}{cccccccccc} 0 & 1 & 0 & 1 & 1 & 0 & 1 & 0 & 1 & 1 \\ 1 & 1 & 0 & 1 & 1 & 1 & 0 & 1 & 1 & 1 \\ \hline ! & \checkmark & \checkmark & \checkmark & \checkmark & ! & ! & ! & \checkmark & \checkmark \end{array}$$

More formally, if we define the **Hamming weight** of a binary string of length n as the number of bits equal to 1, then the Hamming distance between two strings is simply the Hamming weight of their difference (where subtraction is calculated in $\mathbb{Z}/2\mathbb{Z}$, i.e. mod 2). We leave the proof that this is indeed a metric as an exercise (Exercise 12.11.4).

12.2 How far apart are two quantum states?

Given two pure states, $|u\rangle$ and $|v\rangle$, we could try to measure the distance between them using the Euclidean distance $\|u - v\|$. This works for vectors, but has some drawbacks when it comes to quantum states. Recall that a quantum state is not represented by just a unit vector, but by a **ray**, i.e. a unit vector times an arbitrary phase factor. Multiplying a state vector by an overall phase factor has no physical effect: the two unit vectors $|u\rangle$ and $e^{i\phi}|u\rangle$ describe the same state. So, in particular, we want the distance between $|u\rangle$ and $-|u\rangle$ to be zero, since these describe the same quantum state. But if we were to use the Euclidean distance, then we would have that $\|u - (-u)\| = \|u + u\| = 2$, which is actually as far apart as the two unit vectors can be!

One solution to this problem is to define the distance between $|u\rangle$ and $|v\rangle$ as the *minimum* over all phase factors, i.e.

$$d(u, v) := \min_{\phi \in [0, 2\pi)} \left\{ \|u - e^{i\phi}v\| \right\}.$$

You can think of this as just a set, but we have already seen that this is actually a vector space over $\mathbb{Z}/2\mathbb{Z}$, where addition corresponds to XOR.

But with some algebraic manipulation we can actually figure out what this minimum is without calculating any of the other values.

We first express the square of the distance between any two vectors in terms of their inner product:

$$\begin{aligned}\|u - v\|^2 &= \langle u - v | u - v \rangle \\ &= \langle u | u \rangle - \langle u | v \rangle - \langle v | u \rangle + \langle v | v \rangle \\ &= \|u\|^2 + \|v\|^2 - 2 \operatorname{Re} \langle u | v \rangle\end{aligned}$$

(where $\operatorname{Re}(z)$ is the real part of the complex number z). Then we can write the Euclidean distance between state vectors as

$$\|u - v\| = \sqrt{2(1 - \operatorname{Re} \langle u | v \rangle)}.$$

Now if we want to minimise this expression over all rotations of v , then we want $\langle u | v \rangle$ to be real and as large as possible, i.e. for $\langle u | v \rangle = |\langle u | v \rangle|$. This gives us a definition of distance.

The **state distance** between two state vectors $|u\rangle$ and $|v\rangle$ is

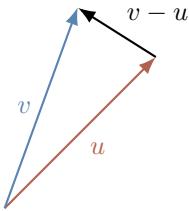
$$d(u, v) := \sqrt{2(1 - |\langle u | v \rangle|)}.$$

Recall that multiplication by a complex number corresponds to rotation and scaling, and so multiplication by a phase factor (which is always of unit length) corresponds to just rotation.

Note that we sometimes write the state distance as $\|u - v\|$, and we might refer to it as “Euclidean distance”, which is an abuse of notation: really we should be writing $\min\{\|u - e^{i\varphi}|v\rangle\|\}$. But this sort of thing happens a lot in mathematics, and it’s good to get used to it. The justification is that, as we have already said, the usual Euclidean distance doesn’t really make great sense for state vectors (because of this vector vs. ray distinction), and so if we know that $|u\rangle$ and $|v\rangle$ are state vectors then writing $\|u - v\|$ (which is already shorthand for $\|u - |v\rangle\|$) should suggest “oh, they mean the version of $\|\cdot\|$ that makes sense for state vectors, where we take a minimum”.

In computer science lingo, this is what you might call **operator overloading**.

For small values of $d(u, v) = \|u - v\|$, we can think of this distance as being the angle between the two unit vectors. Indeed, if we think of Euclidean (unit) vectors, then the difference $v - u$ is, for sufficiently small $\|u - v\|$, just the angle between the two unit vectors (expressed in radians), because a small segment of a circle “almost” looks like a triangle.



Alternatively (and more formally), we can see this by writing $|\langle u | v \rangle| = \cos \alpha \approx 1 - \alpha^2/2$, since then

$$\|u - v\| = \sqrt{2(1 - |\langle u | v \rangle|)} \approx \alpha.$$

This can certainly help with intuition, but extra care must always be taken when dealing with complex vector spaces, since our geometric intuition breaks down rapidly in (complex) dimension higher than 1.

As you might hope, two state vectors which are close to one another give similar statistical predictions. In order to see this, pick a measurement (any measurement)

and consider one partial outcome described by a projector $|a\rangle\langle a|$. What can we say about the difference between the two probabilities

$$p_u = |\langle a|u\rangle|^2$$

$$p_v = |\langle a|v\rangle|^2$$

if we know that $\|u - v\| \leq \varepsilon$?

Well, first of all, let us introduce two classic tricks that are almost always useful when dealing with inequalities — the first holds in any normed vector space, and the latter in any inner product space.

- the **reverse triangle inequality**:

$$|\|u\| - \|v\|| \leq \|u - v\|$$

- the **Cauchy–Schwartz inequality**:

$$\langle u|v\rangle^2 \leq \langle u|u\rangle\langle v|v\rangle$$

or, equivalently (by taking square roots),

$$|\langle u|v\rangle| \leq \|u\|\|v\|.$$

This is arguably *the most useful* mathematical inequality that we have!

Furthermore, the two sides of the inequality are equal *if and only if* $|u\rangle$ and $|v\rangle$ are linearly dependent.

Using these, we see that

$$\begin{aligned} |p_u - p_v| &= \left| |\langle a|u\rangle|^2 - |\langle a|v\rangle|^2 \right| \\ &= \left| \left(|\langle a|u\rangle| + |\langle a|v\rangle| \right) \left(|\langle a|u\rangle| - |\langle a|v\rangle| \right) \right| \\ &\leq 2 \left| |\langle a|u\rangle| - |\langle a|v\rangle| \right| \\ &\leq 2 \left| \langle a|u\rangle - \langle a|v\rangle \right| \\ &\leq 2\|a\|\|u - v\| \\ &= 2\|u - v\|. \end{aligned}$$

So if $\|u - v\| \leq \varepsilon$, then $|p_u - p_v| \leq 2\varepsilon$.

Again, we can appeal to some geometric intuition if we pretend that $|u\rangle$ and $|v\rangle$ are Euclidean vectors instead of rays. Write

$$|\langle a|u\rangle| = \cos(\alpha)$$

$$|\langle a|v\rangle| = \cos(\alpha + \varepsilon)$$

where ε is the (very small) angle between $|u\rangle$ and $|v\rangle$, whence $\|u - v\| = \varepsilon$. Then

$$\begin{aligned} |\langle a|u\rangle|^2 - |\langle a|v\rangle|^2 &= \cos^2(\alpha) - \cos^2(\alpha + \varepsilon) \\ &\approx \varepsilon \sin(2\alpha) \\ &\leq \varepsilon. \end{aligned}$$

As an interesting exercise, you might try to explain why this approach gives a tighter bound (ε instead of 2ε).

12.3 Fidelity

Sometimes, when quantifying closeness of states, the *inner product* is a more convenient tool than the distance/norm. Analogous to how we define the distance between states $|u\rangle$ and $|v\rangle$ as $d(u, v) = \|u - v\|$, we define the **fidelity** between them as

$$F(u, v) := |\langle u | v \rangle|^2.$$

This is *not* a metric, but it does have some similarly nice properties: for example, $F(u, v) = 1$ when the two states are identical, and $F(u, v) = 0$ when the two states are orthogonal (which means that they are “as different as possible”). Intuitively, we can understand fidelity as the probability that the state $|u\rangle$ (resp. $|v\rangle$) would pass a test for being in state $|v\rangle$ (resp. $|u\rangle$). In other words, if we perform an orthogonal measurement on $|u\rangle$ that has two outcomes (true if the state is $|v\rangle$; false if the state is orthogonal to $|v\rangle$), then the fidelity $F(u, v) = |\langle u | v \rangle|^2$ is exactly the probability that we measure the outcome true.

Recall our definition of state distance:

$$d(u, v) = \sqrt{2(1 - |\langle u | v \rangle|)}$$

This gives us a relation between distance and fidelity: once we know one, we can easily calculate the other. However, everything we have said so far applies only to *pure* states — we will see how the mixed state case is slightly more complicated shortly.

One final remark: as another example of the many inconsistencies in the literature, some authors define $F(u, v)$ to be $|\langle u | v \rangle|$ instead of $|\langle u | v \rangle|^2$. Whenever we say fidelity, we mean the latter: $|\langle u | v \rangle|^2$.

12.4 Approximating unitaries

So now we know a bit about how norms (or metrics, or inner products) can help us to understand distance between state vectors, can we say something similar about quantum evolutions? Say we have unitary operators U and V acting on the same Hilbert space, where U is some “target” unitary that we *want* to implement in a real-life circuit, and V is an “approximate” unitary that we *can actually* implement in practice. We say that V **approximates U with precision ϵ** , or that U and V are **ϵ -close**, if

$$\|U - V\| \leq \epsilon$$

where $\|\cdot\|$ is some norm on unitary matrices (of the same size), which we would want to satisfy the following property: if $\|U - V\|$ is “small”, then U should be hard to distinguish from V when acting on *any* quantum state.

Before defining such a norm, however, we first recall some linear algebra which we briefly touched upon in Exercise 5.14.13. The **singular values** of an operator A are the square roots of the (necessarily non-negative) eigenvalues of the Hermitian operator $A^\dagger A$. If A is *normal* (e.g. a density operator), then its singular values are exactly the absolute values of its eigenvalues. We tend to denote singular values by $s_i(A)$ (or just s_i if it is clear which operator we are talking about), and we write $\sigma(A)$ to mean the set of eigenvalues of A , i.e.

$$\sigma(A) := \{\lambda \in \mathbb{C} \mid \det(A - \lambda \mathbf{1}) = 0\}.$$

This means that

$$\{s_i(A)\} = \{\sqrt{\lambda} \mid \lambda \in \sigma(A)\}.$$

Note that V approximates U with precision ϵ if and only if U approximates V with precision ϵ . Even though we might think of one as being our ideal unitary and the other as being the best feasible real-life implementation that we can achieve, this is only us giving names to things — the definition does not care which way round we think of them.

The **operator norm** (or **spectral norm**) $\|A\|$ of an operator $A \in \mathcal{B}(\mathcal{H})$ is the maximum length of the vector $A|v\rangle$ over all possible normalised vectors $|v\rangle \in \mathcal{H}$, i.e.

$$\|A\| := \max_{|v\rangle \in S_{\mathcal{H}}^1} \{|A|v\rangle|\}$$

(where $S_{\mathcal{H}}^1$ is the unit sphere in \mathcal{H} , i.e. the set of vectors of norm 1). One can show that $\|A\|$ is equal to the largest singular value of A .

If A is *normal* (e.g. a density operator), then

$$\|A\| = \max_{\lambda \in \sigma(A)} \{|\lambda|\}.$$

The operator norm satisfies some very useful properties:

- If A is normal, then $\|A^\dagger\| = \|A\|$
- $\|A \otimes B\| = \|A\| \|B\|$
- If U is unitary, then $\|U\| = 1$
- If $P \neq 0$ is an orthogonal projector, then $\|P\| = 1$
- **Sub-multiplicativity:** $\|AB\| \leq \|A\| \|B\|$.

Proving these properties, along with some others, is a good thing to practise — see Exercise 12.11.5.

Now suppose that some quantum system, initially in state $|\psi\rangle$, evolves according to U or V . Let P be a projector associated with some specific outcome of some measurement that can be performed on the system after either evolution (such as $P = |a\rangle\langle a|$, as in our earlier example). Let p_U (resp. p_V) be the probability of obtaining the corresponding measurement outcome if the operation U (resp. V) was performed. By definition, we see that

$$\begin{aligned} |p_U - p_V| &= \left| \langle \psi | U^\dagger P U | \psi \rangle - \langle \psi | V^\dagger P V | \psi \rangle \right| \\ &= \left| \langle \psi | U^\dagger P(U - V) | \psi \rangle + \langle \psi | (U^\dagger - V^\dagger) P V | \psi \rangle \right| \\ &\leq \left| \langle \psi | U^\dagger P(U - V) | \psi \rangle \right| + \left| \langle \psi | (U^\dagger - V^\dagger) P V | \psi \rangle \right| \end{aligned}$$

where the inequality is exactly the triangle inequality.

By an application of the Cauchy–Schwartz inequality followed by sub-multiplicativity, See Exercise 12.11.5. we then have

$$\begin{aligned} |p_U - p_V| &\leq \|U^\dagger P\| \|U - V\| + \|U^\dagger - V^\dagger\| \|V P\| \\ &\leq 2\|U - V\|. \end{aligned}$$

This tells us what ε -closeness means: suppose that V and U are ε -close; then if, instead of applying one, we apply the other, and subsequently measure the resulting physical system, we know that the probabilities of any particular outcome in any measurement will differ by at most 2ε .

Now what about working with *sequences* of unitaries, as we do when we construct quantum circuits? It turns out that closeness is additive under multiplication of unitaries: if $\|U_1 - V_1\| \leq \varepsilon_1$ and $\|U_2 - V_2\| \leq \varepsilon_2$, then

$$\begin{aligned} \|U_2 U_1 - V_2 V_1\| &= \|U_2 U_1 - V_2 U_1 + V_2 U_1 - V_2 V_1\| \\ &= \|(U_2 - V_2) U_1 + V_2 (U_1 - V_1)\| \\ &\leq \|U_2 - V_2\| \|U_1\| + \|V_2\| \|U_1 - V_1\| \\ &= \|U_2 - V_2\| + \|U_1 - V_1\| \\ &\leq \varepsilon_1 + \varepsilon_2. \end{aligned}$$

We can then apply this argument inductively.

Errors in the approximation of one sequence of unitaries by another accumulate at most linearly in the number of unitary operations:

$$\|U_n \cdots U_1 - V_n \cdots V_1\| \leq \sum_{i=1}^n \varepsilon_i$$

if $\|U_i - V_i\| \leq \varepsilon_i$ for all $i = 1, \dots, n$.

This linear error accumulation relies heavily on the fact that the norm of a unitary operator is equal to 1; for non-unitary operators, errors could accumulate exponentially, which would make efficient approximations of circuits practically impossible. Geometrically, this is because unitaries just *rotate* vectors, without scaling them.

Again, we can appeal to some trigonometry. First note that

$$\|U - V\| = \|UV^\dagger - \mathbf{1}\|$$

since the operator norm is unitarily invariant. Since UV^\dagger is also unitary, its eigenvalues are exactly phase factors $e^{i\varphi}$ for $\varphi \in \mathbb{R}$; the corresponding eigenvalue of $UV^\dagger - \mathbf{1}$ has modulus

$$|e^{i\varphi} - 1| = \sqrt{2\sqrt{1 - \cos \varphi}}.$$

Putting this all together, we see that asking for $\|U - V\| \leq \varepsilon$ is exactly asking for each eigenvalue of $UV^\dagger - \mathbf{1}$ to satisfy $\sqrt{2\sqrt{1 - \cos \varphi}} \leq \varepsilon$, which rearranges to

$$\cos \varphi \geq 1 - \frac{\varepsilon^2}{2}$$

which is simply $|\varphi| \leq \varepsilon$ for small enough ε . So U rotates relative to V by (at worst) an angle of order ε , and if we compose unitaries in a sequence then the accumulated rotation increases linearly with the number of unitaries.

12.5 Approximating generic unitaries is hard, but...

Now that we understand approximations of unitary operators, we can revisit the question of universality that we touched upon in Sections 2.13 and 3.5. Recall that we call a finite set G of gates **universal** if *any* n -qubit unitary operator can be approximated (up to an overall phase) to *arbitrary accuracy* by some unitary constructed using only gates from G (and we then call the gates in G **elementary**). In other words, G is universal if, for any unitary U acting on n -qubits and for any $\varepsilon > 0$, there exist $U_1, \dots, U_d \in G$ such that $\tilde{U} := U_d \cdots U_1$ satisfies

$$\|\tilde{U} - e^{i\varphi} U\| \leq \varepsilon$$

for some phase φ .

For example, each of the following sets of gates is universal:

- $\{H, c-S\}$
- $\{H, T, c-\text{NOT}\}$
- $\{H, S, \text{Toff}\}$

where S and T are the $\pi/4$ - and $\pi/8$ -phase gates (Section 2.6), $c-S$ is the *controlled S* gate, and Toff is the Toffoli gate (Exercise 9.12.14).

But now we can be a bit more precise with the question that the notion of universality is trying to answer: given a universal set of gates, how hard is it to approximate any desired unitary transformation with accuracy ε ? That is, *how many gates do we need?*

See Exercise 12.11.5.

The answer is *a lot*. In fact, it is exponential in the number of qubits — most unitary transformation require large quantum circuits of elementary gates. We can show this by a counting argument (along with a healthy dose of geometric intuition).

Consider a universal set of gates G consisting of g gates, where each gate acts on no more than k qubits. How many circuits (acting on n -qubits) can we construct using t gates from this set? We have $g \binom{n}{k}$ choices for the first gate, since there are g gates, and $\binom{n}{k}$ ways to place it so that it acts on k out of n qubits. The same holds for all subsequent gates, and so we can build no more than

$$\left(g \binom{n}{k} \right)^t$$

circuits of size t from G . What is important is that $g \binom{n}{k}$ is *polynomial* in n , and g and k are fixed constants, so we will write this upper bound as

$$(\text{poly}(n))^t.$$

In more geometric language, we have shown that, with t gates, we can generate $(\text{poly}(n))^t$ points in the space $U(N)$ of unitary transformations on n -qubits, where $N = 2^n$. Now imagine drawing a ball of radius ε (in the operator norm) centred at each of these points — we want these balls to cover the entire unitary group $U(N)$, since this then says that any unitary is within distance ε of a circuit built from t gates in G . We will not get into the details of the geometry of $U(N)$, but simply use the fact that a ball of radius ε in $U(N)$ has volume proportional to ε^{N^2} , whereas the volume of $U(N)$ itself is proportional to C^{N^2} for some fixed constant C . So we want

$$\varepsilon^{N^2} (\text{poly}(n))^t \geq C^{N^2}$$

which (after some algebraic manipulation) requires that

$$t \geq 2^{2n} \frac{\log(C/\varepsilon)}{\log(\text{poly}(n))}.$$

In words, *the scaling is exponential in n but only logarithmic in $1/\varepsilon$* .

When we add qubits, the space of possible unitary operations grows very rapidly, and we have to work exponentially hard if we want to approximate the resulting unitaries with some prescribed precision. If, however, we fix the number of qubits and instead ask for better and better approximations, then things are much easier, since we only have to work logarithmically hard.

The snag is that this counting argument does not give us any hints as to how we can actually build such approximations. A more constructive approach is to pick a set of universal gates and play with them, building more and more complex circuits. There is an important theorem in this direction that tells us that it does not matter much which particular universal set of gates we choose to start with.

The Solovay–Kitaev Theorem. Choose any two universal sets of gates that are **weakly closed** under inverses (that is, the inverse of any gate in the set can be constructed (exactly) as a *finite* sequence of gates in the set, even if it is not itself an elementary gate). Then any t -gate circuit built from one set of gates can be implemented to precision ε using a $t \text{poly}(\log(t/\varepsilon))$ -gate circuit built from the other set. Furthermore, there is an efficient classical algorithm for finding this circuit.

Counting arguments nearly always use **binomial coefficient notation**: $\binom{a}{b} := \frac{a!}{b!(a-b)!}$.

Since errors accumulate linearly, it suffices to approximate each gate from one set to accuracy ε/t , which can be achieved by using a $\text{poly}(\log(t/\varepsilon))$ -gate circuit built from the other set. So we can efficiently convert (constructively, via some efficient classical algorithm) between universal sets of gates with overhead $\text{poly}(\log(1/\varepsilon))$, i.e. $\log^c(1/\varepsilon)$ for some constant c . For all practical purposes, we want to minimise c , but the counting argument above shows that the best possible exponent is 1, so the real question is *can we get close to this lower bound?* In general, we do not know. However, for some universal sets of gates we have *nearly* optimal constructions. For example, the set $\{H, T\}$ can be used to approximate arbitrary *single-qubit* unitaries to accuracy ε using $\log(1/\varepsilon)$ many gates, instead of $\text{poly}(\log(1/\varepsilon))$, and the circuits achieving this improved overhead cost can be efficiently constructed (for example, by the [Matsumoto–Amano construction](#)).

12.6 How far apart are two probability distributions?

Before we switch gears and discuss how to generalise state distance to density operators, let us first take a look at distances between probability distributions. What does it mean to say that two probability distributions (over the same index set) are similar to one another?

Recall that a **probability distribution** consists of two things — a **sample space** Ω , which is the set of all possible outcomes, and a **probability function** $p: \Omega \rightarrow [0, 1]$, which tells us the probability of any specific outcome — subject to the condition that $\sum_{k \in \Omega} p(k) = 1$. Given any subset of outcomes $A \subseteq \Omega$, we define $p(A) = \sum\{k \in A\}p(k)$.

Things are simpler for us because we work with so-called **discrete** probability distributions, and so we can use *sums* instead of *integrals*. The general theory requires much more real analysis.

The **trace distance** (also known as the **variation** distance, L_1 distance, **statistical** distance, or **Kolmogorov** distance) between probability distributions p and q on the same sample space Ω is

$$d(p, q) := \frac{1}{2} \sum_{k \in \Omega} |p(k) - q(k)|.$$

This is indeed a distance: it satisfies all the necessary properties. It also has a rather simple interpretation, as we now explain. Let $p(k)$ be the *intended* probability distribution of an outcome produced by some ideal device P , but suppose that the actual physical device Q is slightly faulty: with probability $1 - \varepsilon$ it works exactly as P does, but with probability ε it goes completely wrong and generates an output according to some arbitrary probability distribution $e(k)$. What can we say about the probability distribution $q(k)$ of the outcome of such a device? Well, we can exactly say that

$$d(p, q) \leq \varepsilon$$

by substituting $q(k) = (1 - \varepsilon)p(k) + \varepsilon e(k)$. Conversely, if $d(p, q) = \varepsilon$ then we can represent one of them (say, $q(k)$) as the probability distribution resulting from a process that generates outcomes according to $p(k)$ followed by a process that alters outcome k with total probability not greater than ε .

Note that the normalisation property of probabilities implies that

$$\sum_k p(k) - q(k) = 0.$$

We can split up this sum into two parts: the sum over k for which $p(k) \geq q(k)$, and the sum over k for which $p(k) < q(k)$. If we call the first part S , then the fact that

Exercise. Show that this distance satisfies the triangle inequality.

$\sum_k p(k) - q(k) = 0$ tells us that the second part must be equal to $-S$. Thus

$$\sum_k |p(k) - q(k)| = S + |-S| = 2S$$

whence $S = d(p, q)$.

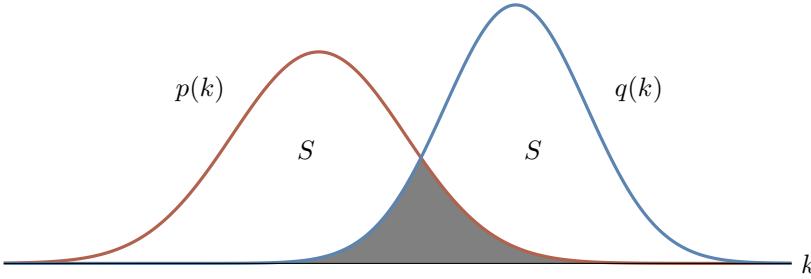


Figure 12.1: Visualising the distance between two (continuous) probability distributions.

Just as a passing note, we will point out that

$$\begin{aligned} \sum_k \max\{p(k), q(k)\} &= 1 + S \\ &= 1 + d(p, q) \end{aligned}$$

and the shaded area in Figure 12.1 is equal to

$$\begin{aligned} \sum_k \min\{p(k), q(k)\} &= 1 - S \\ &= 1 - d(p, q). \end{aligned}$$

The latter lets us write the trace distance as

$$d(p, q) = 1 - \sum_k \min\{p(k), q(k)\}$$

and the former will be useful very soon.

As for intuition, the trace distance is a measure of how well we can distinguish a sample from distribution p from a sample from distribution q : if the distance is 1 then we can tell them apart perfectly; if the distance is 0 then we can't distinguish them at all. Now suppose that p and q represent the probability distributions of two devices, P and Q , respectively, and that one of these is chosen (with equal probability) to generate some outcome. If you are given the outcome k , and you know $p(k)$ and $q(k)$, then how can you best guess which device generated it? What is your best strategy, and with what probability does this let you guess correctly? It turns out that we can answer this using the trace distance.

Arguably the most natural strategy is to look at $\max\{p(k), q(k)\}$: guess P if $p(k) > q(k)$; guess Q if $q(k) > p(k)$; guess uniformly at random if $p(k) = q(k)$. Following this strategy, the probability of guessing correctly (again, under the assumption that P and Q were chosen between with equal probability) is

$$p_{\text{success}} = \frac{1}{2} \sum_k \max\{p(k), q(k)\}$$

which we can rewrite as

$$p_{\text{success}} = \frac{1}{2}(1 + d(p, q)).$$

Again, since we are working with finite probability distributions, we can use sums; in the continuous case shown in Figure 12.1, we would really need to use *integrals* instead.

Here is another way of seeing the above. The probability that the devices P and Q will *not* behave in the same way is bounded by $d(p, q)$. This means that, with probability $1 - d(p, q)$, the devices behave as if they were identical, in which case the best you can do is to guess uniformly at random, which will make you succeed with probability $\frac{1}{2}(1 - d(p, q))$. With the remaining probability $d(p, q)$, the devices may behave as if they were completely different, and then you can tell which one is which perfectly, letting you succeed with probability exactly $1 \cdot d(p, q) = d(p, q)$. So the total probability of success is equal to

$$\frac{1}{2}(1 - d(p, q)) + d(p, q) = \frac{1}{2}(1 + d(p, q)).$$

12.7 Dealing with density operators

Now we return to quantum states, and generalise the notion of trace distance to density operators.

The **trace norm** of an operator is the sum of its singular values:

$$\|A\|_{\text{tr}} := \sum_i s_i(A).$$

If A is *normal* (e.g. a density operator), then

$$\|A\|_{\text{tr}} = \sum_{\lambda \in \sigma(A)} |\lambda|.$$

The induced **trace distance** between two density operators is

$$d_{\text{tr}}(\rho, \sigma) := \frac{1}{2} \|\rho - \sigma\|_{\text{tr}}.$$

There are many questions raised by this definition, such as “how does this relate to the trace distance of probability distributions?” and “how does this trace norm relate to the operator norm from Section 12.4?” — we will answer the first question now, but our answer to the second builds upon the notion of an ℓ^p -norm, which is a discussion that we will postpone for Section 12.11.2.

We can simply think of the trace distance for density operators as the natural analogue of the trace distance for probability distributions: it is a tight upper bound on the distances between the probability distributions obtained from ρ and σ by a measurement, as we now justify.

Let $\{P_k\}$ be a complete set of orthogonal projectors, defining a projective measurement in some \mathcal{H} . This measurement gives outcome k with some probability $p(k)$ if the quantum system is in state ρ , and the same outcome with some probability $q(k)$ if the system is in state σ . That is,

$$\begin{aligned} p(k) &:= \text{tr } P_k \rho \\ q(k) &:= \text{tr } P_k \sigma. \end{aligned}$$

Then

$$\begin{aligned} d_{\text{tr}}(p, q) &:= \frac{1}{2} \sum_k |p(k) - q(k)| \\ &= \frac{1}{2} \sum_k |\text{tr } P_k (\rho - \sigma)| \\ &= \frac{1}{2} \text{tr}((\rho - \sigma) U) \end{aligned}$$

where we define

$$U := \sum_k \frac{\text{tr } P_k(\rho - \sigma)}{|\text{tr } P_k(\rho - \sigma)|} P_k$$

or, in other words, U is the sum of the P_k but where the signs are determined by whether $|\text{tr } P_k(\rho - \sigma)|$ is equal to $+\text{tr } P_k(\rho - \sigma)$ or $-\text{tr } P_k(\rho - \sigma)$.

Since this U is unitary, and since the trace norm can be written as

See Section 12.11.2.

$$\|A\|_{\text{tr}} = \max_{U \text{ unitary}} |\text{tr } AU|$$

we finally obtain that

$$\begin{aligned} d_{\text{tr}}(p, q) &:= \frac{1}{2} \sum_k |p(k) - q(k)| \\ &= \leq \frac{1}{2} \|\rho - \sigma\| \\ &=: d_{\text{tr}}(\rho, \sigma) \end{aligned}$$

which says that the trace distance $d_{\text{tr}}(\rho, \sigma)$ gives an upper bound on distances between probability distributions obtained from ρ and σ by a measurement. The fact that this bound is tight (i.e. attainable) is witnessed by the measurement defined by the projectors onto the eigenspaces of $\rho - \sigma$.

As an example, consider pure states $|u\rangle$ and $|v\rangle$. The trace distance between them is

$$\frac{1}{2} \||u\rangle\langle u| - |v\rangle\langle v|\|_{\text{tr}}.$$

We can write $|v\rangle$ as

$$|v\rangle = \alpha|u\rangle + \beta|\bar{u}\rangle$$

where $|\bar{u}\rangle$ is some unit vector orthogonal to $|u\rangle$, and where $\alpha = \langle u|v\rangle$, with β determined by $|\alpha|^2 + |\beta|^2 = 1$. Then

$$\begin{aligned} |u\rangle\langle u| - |v\rangle\langle v| &= \begin{bmatrix} 1 & 0 \\ 0 & 0 \end{bmatrix} - \begin{bmatrix} |\alpha|^2 & \alpha\beta^* \\ \alpha^*\beta & |\beta|^2 \end{bmatrix} \\ &= \begin{bmatrix} |\beta|^2 & -\alpha\beta^* \\ -\alpha^*\beta & -|\beta|^2 \end{bmatrix} \end{aligned}$$

(which has eigenvalues $\pm|\beta|$), and the trace distance is given by

$$\frac{1}{2} \||u\rangle\langle u| - |v\rangle\langle v|\|_{\text{tr}} = \sqrt{1 - |\langle u|v\rangle|^2}$$

which is exactly $\sqrt{1 - \text{fidelity}}$.

As a consequence of this, we see that

$$\frac{1}{2} \||u\rangle\langle u| - |v\rangle\langle v|\|_{\text{tr}} \leq \|u - v\|$$

since

$$\begin{aligned} 1 - |\langle u|v\rangle|^2 &= (1 + |\langle u|v\rangle|)(1 - |\langle u|v\rangle|) \\ &\leq 2(1 - |\langle u|v\rangle|) \\ &= \|u - v\|^2. \end{aligned}$$

So if two states $|u\rangle$ and $|v\rangle$ are ε -close in the trace distance, then the probability distributions of outcomes of *any* measurement performed on a physical system in state $|u\rangle$ or $|v\rangle$ will also be ε -close in the trace distance.

12.8 Distinguishing non-orthogonal states, again

Let's briefly return to the problem considered in Section 4.9, where we are given a system and told that it is in either state $|\psi_1\rangle$ or $|\psi_2\rangle$, with equal probability, but that these two vectors are *not* orthogonal. The goal is to find a measurement that maximises the probability of correctly identifying which state the system is in. Before solving this problem using the language of distances, let us repeat the geometric idea that we used previously.

Draw two vectors, $|\psi_1\rangle$ and $|\psi_2\rangle$, separated by some angle ε . We want to find some orthonormal vectors $|e_1\rangle$ and $|e_2\rangle$ that specify the optimal projective measurement. First, note that any projections on the subspace *orthogonal* to the plane spanned by $|\psi_1\rangle$ and $|\psi_2\rangle$ will reveal no information about the identity of the state, so we know that we will want our orthonormal vectors to lie in the span of $|\psi_1\rangle$ and $|\psi_2\rangle$. Now, since $|\psi_1\rangle$ and $|\psi_2\rangle$ are both equally likely to occur, we want to place $|e_1\rangle$ and $|e_2\rangle$ *symmetrically* around them, as shown in Figure 12.2

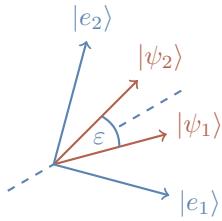


Figure 12.2: Recall Section 4.9 — the optimal measurement to distinguish between the two equally likely non-orthogonal signal states $|\psi_1\rangle$ and $|\psi_2\rangle$ is described by the two orthogonal vectors $|e_1\rangle$ and $|e_2\rangle$ placed symmetrically around them.

The probability of correctly distinguishing the two states is

$$p_{\text{success}} = \frac{1}{2}|\langle e_1|\psi_1\rangle|^2 + \frac{1}{2}|\langle e_2|\psi_2\rangle|^2$$

which reduces, with our schema, to

$$\begin{aligned} p_{\text{success}} &= \cos^2\left(\frac{\pi}{2} - \frac{\varepsilon}{2}\right) \\ &= \frac{1}{2}(1 + \sin \varepsilon) \\ &\approx \frac{1}{2}(1 + \varepsilon) \\ &= \frac{1}{2}(1 + \|\psi_1 - \psi_2\|). \end{aligned}$$

where the last equality holds whenever ε is “small enough”. But, happily, we can be much more precise than this!

Let's start by rephrasing the problem in terms of density operators. We are sent one of two quantum states, either ρ_0 or ρ_1 , with equal probability. You might notice that we're now labelling our states with $\{0, 1\}$ instead of $\{1, 2\}$. This is simply to help guide our intuition: we are being sent one bit of information, a 0 or a 1; the only complication is that this is happening in such a way that we cannot perfectly distinguish between them (since we are receiving *non-orthogonal* quantum states). We want to choose two orthogonal projectors P_0 and P_1 , so outcome P_0 is interpreted as a 0 and outcome P_1 as a 1. The probability of correctly detecting which state was sent is, as always, the probability that ρ_0 was sent *and* outcome P_0 was observed, plus

the probability that ρ_1 was sent *and* outcome P_1 was observed. In symbols,

$$\begin{aligned} p_{\text{success}} &= \frac{1}{2} \text{tr}(P_0\rho_0) + \frac{1}{2} \text{tr}(P_1\rho_1) \\ &= \frac{1}{4} \text{tr}[(P_0 + P_1)(\rho_0 + \rho_1)] + \frac{1}{4} \text{tr}[(P_0 - P_1)(\rho_0 - \rho_1)] \\ &= \frac{1}{2} + \frac{1}{4} \text{tr}[(P_0 - P_1)(\rho_0 - \rho_1)] \end{aligned}$$

where the last equality follows from the fact that $P_0 + P_1 = \mathbf{1}$.

By applying Hölder's inequality, this tells us that

$$\begin{aligned} p_{\text{success}} &\leq \frac{1}{2} + \frac{1}{4} \|(P_0 - P_1)\| \|(\rho_0 - \rho_1)\|_{\text{tr}} \\ &\leq \frac{1}{2} + \frac{1}{4} \|\rho_0 - \rho_1\|_{\text{tr}} \\ &= \frac{1}{2}(1 + d_{\text{tr}}(\rho_0, \rho_1)). \end{aligned}$$

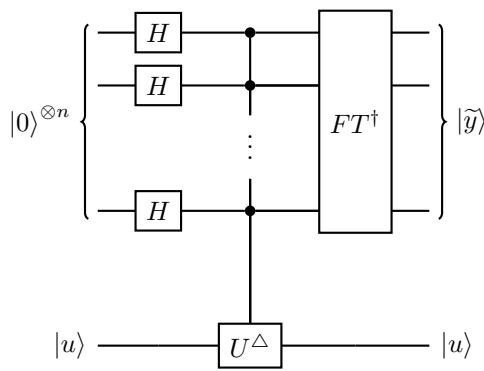
Again, this upper bound is attained by taking P_0 to be the projector onto the eigenspace of $\rho_0 - \rho_1$ corresponding to the *positive* eigenvalues, and P_1 the projector corresponding to the *negative* eigenvalues: this gives $\text{tr}(P_0 - P_1)(\rho_0 - \rho_1) = \|\rho_0 - \rho_1\|_{\text{tr}}$.

Of course, the whole story of quantum state distinguishability has much more to it than we have covered here. In Exercise 12.11.9 we ask about the case where the two states ρ_0 and ρ_1 are sent with *non-equal* probabilities p_0 and p_1 , respectively. The more general scenario, where some quantum source emits states ρ_0, \dots, ρ_n with respective probabilities p_0, \dots, p_n , turns out to be incredibly difficult — we do not know an optimal discrimination strategy, except for in a few special cases.

12.9 Approximate phase estimation

In Section 10.8 we showed how to determine the phase of an eigenvalue of a controlled- U gate, hidden by an oracle, under the assumption that the phase was of a particularly nice rational form: $2\pi m/2^n$. More precisely, we were able to find the integer $m \bmod 2^n$. Here we will improve upon this result, explaining how to adapt the same circuit for arbitrary phases.

So say we have some controlled- U gate with eigenvector $|u\rangle$ and eigenvalue $e^{i\varphi}$. We can run the same circuit as we did in Section 10.8:



As before, the Hadamard gate followed by the controlled- U^Δ gate prepare the state

$$\frac{1}{\sqrt{N}} \sum_{x=0}^{N-1} e^{i\varphi x} |x\rangle |u\rangle$$

where $N = 2^n$, with n the number of qubits in the first register, since the phase kick-back leaves the eigenstate $|u\rangle$ unchanged. Now we apply the inverse quantum Fourier transform, giving

$$\frac{1}{N} \sum_{x,y=0}^{N-1} e^{ix(\varphi - 2\pi y/N)} |y\rangle |u\rangle.$$

We already know that if $N\varphi/2\pi$ has an exact n -bit representation (i.e. if $\varphi = 2\pi m/2^n$ with $0 \leq m < N$) then we are guaranteed to recover this when we measure the output $|\tilde{y}\rangle$.

If this is not the case, then instead we can only hope for $|\tilde{y}\rangle$ to be the *best n-bit approximation* to $N\varphi/2\pi$. This means that the distance between the two can be no more than $1/2$ (otherwise $|\tilde{y}\rangle$ would round to $N\varphi/2\pi \pm 1$ instead). Rearranging this inequality gives

$$\left| \varphi - \frac{2\pi\tilde{y}}{N} \right| \leq \frac{\pi}{N}$$

which, if we define $\delta = \varphi - 2\pi\tilde{y}/N$, is exactly $|\delta| \leq \pi/N$. Now we can calculate the probability of measuring the result $|\tilde{y}\rangle$ as

$$\begin{aligned} p &:= \frac{1}{N^2} \left| \sum_{x=0}^{N-1} e^{ix\delta} \right|^2 \\ &= \frac{\sin^2(N\delta/2)}{N^2 \sin^2(\delta/2)}. \end{aligned}$$

Let's see if we can find a lower bound for this probability.

First of all, for small values of θ , we know that $\sin \theta < \theta$. In particular, $\sin(\delta/2) < \delta/2$. Secondly, we can show that $\sin(N\delta/2) > N\delta/\pi$ by using the fact that $N\delta \leq \pi$ (though here we simply provide Figure 12.3 instead of giving a proof). Thus

$$\begin{aligned} p &> \left(\frac{2N\delta}{N\delta\pi} \right)^2 \\ &= \frac{4}{\pi^2} \approx 0.41 \end{aligned}$$

and so we find the best n -bit approximation to φ with pretty good probability.

In fact, the coefficients in these inequalities work in our favour: the further away a result $|\tilde{y}\rangle$ is from being the best n -bit approximation to φ , the lower our probability of measuring it. This provides a way of getting good approximations with even higher probability, and one that we can choose ourselves, by using more qubits. That is, if we increase the number of qubits in the first register to be t , for some $t > n$, then we know that with probability $p = 4/\pi^2$ we will get the best t -bit approximation to φ . But we're only interested in the best n -bit approximation, and the 2^{t-n} next-most-likely outcomes — those that will truncate to give the same n -bit approximation — from the t -qubit circuit are all the next highest probability ones. If one sums everything up carefully, then we can show that the probability of *not* measuring one of these is

$$\varepsilon \leq \frac{1}{2(2^{t-n} - 2)}$$

and so, for any desired ε , we get the best n -bit approximation with probability at least $1 - \varepsilon$ by setting t to be

$$t = n + \left\lceil \log_2 \left(2 + \frac{1}{2\varepsilon} \right) \right\rceil$$

(where $\lceil x \rceil$ denotes the ceiling of x , i.e. the smallest integer larger than x).

This is from the Taylor expansion $\sin \theta \approx \theta - \theta^3/3!$ and some real analysis. However, this is the sort of fact that you will often see quoted without justification, because it's used so often.

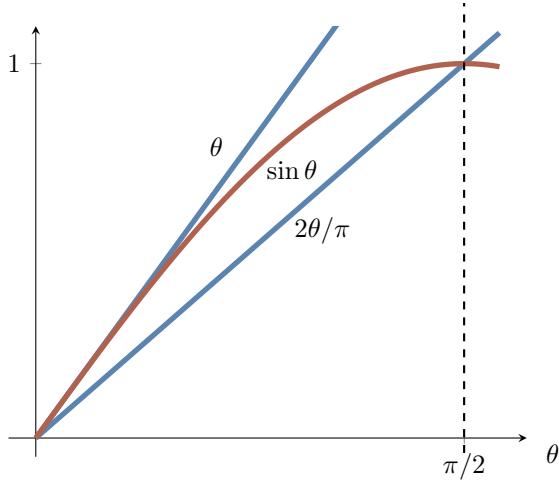


Figure 12.3: In the region $0 \leq \theta \leq \pi/2$, we can bound $\sin \theta$ above and below by the linear functions θ and $2\theta/\pi$, respectively.

12.10 How accurate is accurate enough?

We have seen that finite sets of gates can be used to approximate any unitary operation with any prescribed accuracy. But how accurate is accurate enough? Of course, the answer depends on what we want to achieve.

Suppose we come up with a cool quantum algorithm, represented by a circuit composed of t gates, and it solves an interesting decision problem with probability $\frac{1}{2} + \delta$. The value of δ might be tiny, so not much can be inferred from a single run, but as long as we can repeat the computation r times and take the majority answer as the “right” answer, the Chernoff bound tells us that the probability of error is bounded above by $e^{-2r\delta^2}$. We now want to physically implement this circuit using our preferred universal set of gates, say $\{H, T, \text{c-NOT}\}$. If we can implement each gate with accuracy ε/t , then we can approximate the circuit with accuracy ε , which means that the probability of success will be $\frac{1}{2} + \delta \pm \varepsilon$. So, at the very least, we want $\varepsilon < \delta/2$.

Recall Exercise 1.11.10.

12.11 Remarks and exercises

12.11.1 Operator decompositions

Analogously to how we can factor polynomials into linear parts, or factor numbers into prime divisors, we can “factor” matrices into smaller components. Doing so often helps us to better understand the geometry of the situation: we might be able to understand the transformation described by a single matrix as “some reflection, followed by some rotation, followed by some scaling”. For us, one specific use of such a “factorisation” (known formally as an **operator decomposition**) is in better understanding various operator norms, as we explain in Exercise 12.11.2.

Here are three operator decompositions that are particularly useful in quantum information theory. The second is for arbitrary operators between Hilbert spaces, the first and third are for *normal endomorphisms* (i.e. normal operators from one Hilbert space to itself).

1. **Spectral decomposition.** Recall Section 4.5: the spectral theorem tells us that every *normal* operator $A \in \mathcal{B}(\mathcal{H})$ can be expressed as a linear combination of projections onto pairwise orthogonal subspaces. We write the spectral decom-

position of A as

$$A = \sum_k \lambda_k |v_k\rangle\langle v_k|$$

where λ_k are the eigenvalues of A , with corresponding eigenvectors $|v_k\rangle$, which form an orthonormal basis in \mathcal{H} .

In matrix notation, we can write this as

$$A = UDU^\dagger$$

where D is the diagonal matrix whose diagonal entries are the eigenvalues λ_k , and where U is the unitary matrix whose columns are the eigenvectors $|v_k\rangle$.

2. Singular value decomposition (SVD). We have already mentioned the SVD in Exercise 5.14.13 when discussing the Schmidt decomposition, but we recall the details here. Consider *any* (non-zero) operator $A \in \mathcal{B}(\mathcal{H}, \mathcal{H}')$. From this, we can construct two positive semi-definite operators: $A^\dagger A \in \mathcal{B}(\mathcal{H})$ and $AA^\dagger \in \mathcal{B}(\mathcal{H}')$. These are both normal, and so we can apply the spectral decomposition to both. In particular, if we denote the eigenvalues of $A^\dagger A$ by λ_k , and the corresponding eigenvectors by $|v_k\rangle$, then we see that the vectors

$$|u_k\rangle := \frac{1}{\sqrt{\lambda_k}} A |v_k\rangle$$

form an orthonormal system in \mathcal{H}' (and are, in fact, eigenvectors of AA^\dagger), since

$$\begin{aligned} \langle u_i | u_j \rangle &= \frac{1}{\sqrt{\lambda_i} \sqrt{\lambda_j}} \langle v_i | A^\dagger A | v_j \rangle \\ &= \frac{\lambda_j}{\sqrt{\lambda_i} \sqrt{\lambda_j}} \langle v_i | v_j \rangle \\ &= \delta_{ij}. \end{aligned}$$

We define the **singular values** s_k of A to be the square roots of the eigenvalues of $A^\dagger A$, i.e. $s_k^2 = \lambda_k$. These singular values satisfy

$$A |v_k\rangle = s_k |u_k\rangle$$

by construction, and so we can write

$$A = \sum_k s_k |u_k\rangle\langle v_k|$$

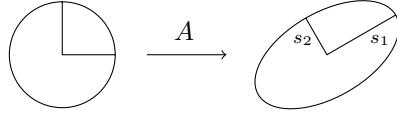
which we call the **singular value decomposition** (or **SVD**). This decomposition holds for arbitrary (non-zero) operators as opposed to just normal ones, and also for operators between two different Hilbert spaces as opposed to just endomorphisms. In words, this decomposition says that, given A , we can find orthonormal bases of \mathcal{H} and \mathcal{H}' such that A maps the k -th basis vector of \mathcal{H} to a non-negative multiple of the k -th basis vector of \mathcal{H}' (and sends any left over basis vectors to 0, if $\dim \mathcal{H} > \dim \mathcal{H}'$).

In matrix notation, we can write this as

$$A = U \sqrt{D} V^\dagger$$

where D is the diagonal matrix of eigenvalues (and so \sqrt{D} is the diagonal matrix of *singular values*), and both U and V are unitary.

Geometrically, we are decomposing any linear transformation into a composition of a rotation or reflection V^\dagger , followed by a scaling by the singular values \sqrt{D} , followed by another rotation or reflection U . This maps the unit sphere in \mathcal{H} onto an ellipsoid in \mathcal{H}' , and the singular values of A are exactly the lengths of the semi-axes of this ellipsoid.



3. **Polar decomposition.** Let $A \in \mathcal{B}(\mathcal{H})$ be a normal arbitrary operator. Since it is an endomorphism, it is represented by a square matrix. Forgetting that A is normal for a moment, we know that its SVD takes the form

$$\begin{aligned} A &= U\sqrt{A^\dagger A} \\ &= \sqrt{AA^\dagger}U \end{aligned}$$

where the unitary matrix U connects the two eigenbases: $U = \sum_k |u_k\rangle\langle v_k|$. We shall return to this unitary U shortly.

Since A is normal, $A^\dagger A = AA^\dagger$, so we can define its **modulus** as

$$|A| := \sqrt{A^\dagger A}$$

which gives us the **polar decomposition**

$$A = |A|U.$$

This is the matrix analogue of the polar decomposition of a complex number: $z = re^{i\theta}$.

If we decompose the eigenvalues of A as $\lambda_k = r_k e^{i\theta_k}$ (with corresponding eigenvectors $|v_k\rangle$) then the spectral decomposition of A gives us

$$\begin{aligned} A &= \sum_k \lambda_k |v_k\rangle\langle v_k| \\ &= \sum_k r_k e^{i\theta_k} |v_k\rangle\langle v_k| \\ &= \sum_k r_k |u_k\rangle\langle v_k| \end{aligned}$$

where $|u_k\rangle = e^{i\theta_k} |v_k\rangle$, so we see that the unitary $U = \sum_k |u_k\rangle\langle v_k|$ in the polar decomposition contains all the information of the phase factors.

12.11.2 More operator norms

We have already seen, all the way back in Section 1.11.2, how the Euclidean norm (from which we get the Euclidean distance) is the special case $p = 2$ of p -norms (also known as ℓ^p -norms), where

$$\|v\|_p := \left(\sum_{i=1}^n |v_i|^p \right)^{\frac{1}{p}}$$

Think how far you've come since then!

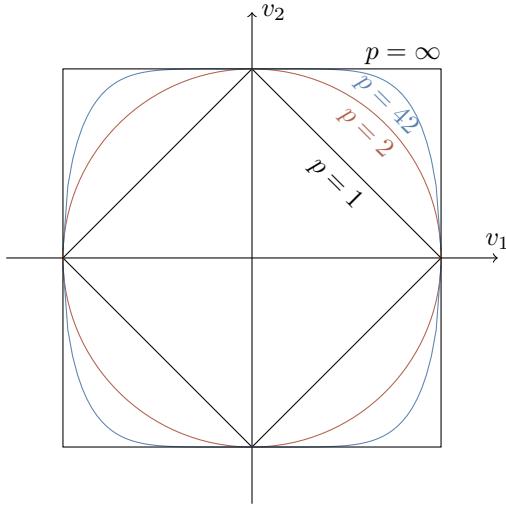
for a vector $v = (v_1, \dots, v_n) \in \mathbb{R}^n$, and for $p \in \mathbb{N}$. We can actually extend this definition to include the case $p = \infty$ by setting

$$\|v\|_\infty := \max_{1 \leq i \leq n} |v_i|.$$

One particularly nice consequence of this definition is that

$$\|v\|_1 \geq \|v\|_2 \geq \dots \geq \|v\|_\infty$$

for any vector v .



You might recall that we named the Cauchy–Schwartz inequality as arguably the most useful inequality in analysis. Well it turns out that it is actually the special case $p = 2$ of an inequality concerning p -norms.

Hölder's inequality. Let $p, q \in \mathbb{N} \cup \{\infty\}$ be such that $\frac{1}{p} + \frac{1}{q} = 1$. Then

$$\|vw\|_1 \leq \|v\|_p \|w\|_q$$

with equality if and only if v and w are “ (p, q) -linearly dependent”.

We will come back to the relevance of these p -norms shortly. For now, let us introduce three norms (two of which we have already seen, but recall here again to tell a more complete story) on the space of endomorphisms $\mathcal{B}(\mathcal{H})$ of a Hilbert space, each of which can be defined neatly in terms of singular values. Note that, if A is normal, then its singular values are exactly the absolute values of its eigenvalues, which usually lets us simplify the definition of the norm.

Throughout, let $A \in \mathcal{B}(\mathcal{H})$, with singular values s_k .

1. **Spectral norm.** This one is so frequently used that it is often simply called the **operator** norm and denoted simply by $\|\cdot\|$. It is the maximum length of the vector $A|v\rangle$ over all possible normalised vectors $|v\rangle \in \mathcal{H}$, i.e.

$$\|A\| := \max_{|v\rangle \in S_{\mathcal{H}}^1} \{|A|v\rangle\}$$

(where $S_{\mathcal{H}}^1$ is the unit sphere in \mathcal{H} , i.e. the set of vectors of norm 1). From this definition, one can actually show that the norm is given by the largest singular value:

$$\|A\| = \max_k s_k.$$

2. **Trace norm.** This is given by the sum of the singular values of A , i.e.

$$\|A\|_{\text{tr}} := \sum_k s_k$$

but note that we can rewrite this using the polar decomposition (from Section 12.11.1) as simply

$$\|A\|_{\text{tr}} = \text{tr}|A|.$$

3. **Frobenius norm.** We have mentioned a few times how inner products give rise to norms, and you might remember that we introduced an inner product on $\mathcal{B}(\mathcal{H})$ a while ago: the Hilbert–Schmidt norm

$$\begin{aligned}(A|B) &:= \text{tr } A^\dagger B \\ &= \sum_{i,j} A_{ij}^* B_{ji}.\end{aligned}$$

Here we drop the factor of $\frac{1}{2}$ that we sometimes included for simplifying certain calculations.

The Frobenius norm is the norm induced by this inner product, i.e.

$$\begin{aligned}\|A\|_F &:= \sqrt{(A|A)} \\ &= \sqrt{\text{tr}(A^\dagger A)} \\ &= \sqrt{\sum_{i,j} |A_{ij}|^2}.\end{aligned}$$

Let's study the relation between the operator norm and the trace norm first. By definition, we see that

$$\|A\|_{\text{tr}} \geq \|A\|$$

but there is another, more subtle, inequality that they satisfy, namely

$$|(A|B)| \leq \|A\|_{\text{tr}} \|B\|$$

which is like a more general version of the Cauchy–Schwartz inequality, and is sometimes referred to as **Hölder's inequality for matrices**. To derive this inequality, we can use the SVD of A , since then

$$\begin{aligned}|(A|B)| &= |\text{tr } A^\dagger B| \\ &= \left| \text{tr} \left(\sum_k s_k |v_k\rangle \langle u_k| B \right) \right| \\ &= \left| \sum_k s_k \langle u_k | B | v_k \rangle \right| \\ &\leq \sum_k s_k |\langle u_k | B | v_k \rangle| \\ &\leq \sum_k s_k \|B\| \\ &= \|A\|_{\text{tr}} \|B\|.\end{aligned}$$

We can actually use this inequality to recover either the operator or the trace norm, by maximising: for any fixed A (or any fixed B), we can obtain equality:

$$\begin{aligned}\|A\|_{\text{tr}} &= \max_{\|B\|=1} \{(A|B)\} \\ \|B\| &= \max_{\|A\|_{\text{tr}}=1} \{(A|B)\}.\end{aligned}$$

To see this, in the first case we can use the polar decomposition $A = |A|U$ to see that equality is attained by $B = U$; in the second case we can use the polar decomposition $B = |B|V$ to see that equality is attained by $A = V|v_1\rangle\langle v_1|$, where $|v_1\rangle$ is the eigenvector of $|B|$ with the largest eigenvalue (or, equivalently, singular value). In particular, this gives us a **variational** characterisation of the trace norm which is very useful at times:

$$\|A\|_{\text{tr}} = \max_{U \text{ unitary}} |\text{tr } AU|.$$

One final special case to point out is what happens if A is Hermitian. Then we can separate the spectral decomposition into positive and negative parts, and write A as the difference of two positive operators:

$$A = A_+ - A_-.$$

Then

$$\begin{aligned} |A| &= A_+ + A_- \\ \|A\|_{\text{tr}} &= \text{tr } A_+ + \text{tr } A_-. \end{aligned}$$

Now we finally return to the relevance of p -norms: it turns out that all these three operator norms above are actually special cases of [Schatten \$p\$ -norms](#). For $p \in \mathbb{N}$ we define the Schatten p -norm in terms of singular values s_k as

$$\|A\|_p := \left(\sum_k |s_k|^p \right)^{\frac{1}{p}}$$

and we define

$$\|A\|_\infty := \max_k |s_k|$$

analogously to how we did for ℓ^p -norms. We then recover the trace norm, the Frobenius norm, and the spectral norm by taking $p = 1, 2, \infty$, respectively:

$$\begin{aligned} \|A\|_1 &= \|A\|_{\text{tr}} \\ \|A\|_2 &= \|A\|_F \\ \|A\|_\infty &= \|A\|. \end{aligned}$$

All Schatten p -norms are sub-multiplicative ($\|AB\|_p \leq \|A\|_p \|B\|_p$) and unitarily invariant ($\|A\| = \|UAV\|$ for any unitaries U and V).

12.11.3 Fidelity in a trace norm inequality

There is a useful inequality involving the trace norm:

$$\text{tr}(A - B)^2 \leq \|A^2 - B^2\|_{\text{tr}}.$$

Let's prove it!

Let λ_k be the eigenvalues of the operator $A - B$, with corresponding eigenvectors and $|u_k\rangle$. Then

$$\text{tr}(A - B)^2 = \sum_k \lambda_k^2$$

since the trace is exactly the sum of eigenvalues. Now, for any unitary U , we have that

$$\|A^2 - B^2\|_{\text{tr}} \geq |\text{tr}(A^2 - B^2)U|$$

since $\|X\|_{\text{tr}} = \max_{U \text{ unitary}} |\text{tr}(XU)|$. If we take $U = \sum_i \pm |u_k\rangle\langle u_k|$, where the signs are chosen so that each term $\langle u_k | A^2 - B^2 | u_k \rangle$ is non-negative, then

$$\begin{aligned} \|A^2 - B^2\|_{\text{tr}} &\geq |\text{tr}(A^2 - B^2)U| \\ &= \sum_k |\langle u_k | A^2 - B^2 | u_k \rangle|. \end{aligned}$$

Writing $A^2 - B^2$ as

This is the non-commutative version of the identity $a^2 - b^2 = (a + b)(a - b)$.

$$A^2 - B^2 = \frac{1}{2} \left[(A+B)(A-B) + (A-B)(A+B) \right]$$

we see that

$$\begin{aligned} \langle u_k | A^2 - B^2 | u_k \rangle &= \frac{1}{2} \langle u_k | (A+B)(A-B) + (A-B)(A+B) | u_k \rangle \\ &= \lambda_i \langle u_k | A+B | u_k \rangle. \end{aligned}$$

Then, since $\lambda_i = \langle u_k | A | u_k \rangle - \langle u_k | B | u_k \rangle$ is the difference of two non-negative numbers, we have that

$$\langle u_k | A+B | u_k \rangle \geq |\lambda_i|.$$

Putting this all together, we obtain

$$\begin{aligned} \|A^2 - B^2\|_{\text{tr}} &\geq \sum_k |\langle u_k | A^2 - B^2 | u_k \rangle| \\ &\geq \sum_k \lambda_k^2 \\ &= \text{tr}(A-B)^2 \end{aligned}$$

as desired.

One particularly nice application of this inequality arises when we take $A = \sqrt{\rho}$ and $B = \sqrt{\sigma}$, since then

$$\text{tr}(\sqrt{\rho} - \sqrt{\sigma})^2 \leq \|\rho - \sigma\|_{\text{tr}}$$

or, after some rearrangement,

$$1 - \text{tr}(\sqrt{\rho}\sqrt{\sigma}) \leq \frac{1}{2} \|\rho - \sigma\|_{\text{tr}}$$

and we recognise $\text{tr}(\sqrt{\rho}\sqrt{\sigma})$ as the fidelity.

12.11.4 Hamming distance

Show that the Hamming distance (defined in Section 12.1) is indeed a metric.

12.11.5 Operator norm

Prove the following properties of the operator norm:

1. $\|A \otimes B\| = \|A\| \|B\|$ for any operators A and B
2. If A is normal, then $\|A^\dagger\| = \|A\|$
3. If U is unitary, then $\|U\| = 1$
4. If $P \neq 0$ is an orthogonal projector, then $\|P\| = 1$.

Using the singular value decomposition, or otherwise, prove that the operator norm has the following two properties for any operators A and B :

5. **Unitary invariance:** $\|UAV\| = \|A\|$ for any unitaries U and V
6. **Sub-multiplicativity:** $\|AB\| \leq \|A\| \|B\|$.

Recall that we say that V approximates U with precision ε if $\|U - V\| \leq \varepsilon$.

Recall Exercise 5.14.13

7. Prove that, if V approximates U with precision ε , then V^{-1} approximates U^{-1} with the same precision ε .

Using the Cauchy–Schwartz inequality, or otherwise, prove the following, for any vector $|\psi\rangle$ and any operators A and B :

8. $|\langle \psi | A^\dagger B | \psi \rangle| \leq \|A\| \|B\|$.

12.11.6 Tolerance and precision

Suppose we wish to implement a quantum circuit consisting of gates U_1, \dots, U_d , but we only have available to us gates V_1, \dots, V_d . Luckily, these gates happen to be pretty good approximations to our desired gates, and the error is uniform: $\|U_i - V_i\| \leq \varepsilon$ for all $i = 1, \dots, d$ for some fixed ε .

We want our approximate circuit to be within some **tolerance** δ of the desired circuit: the probabilities of different outcomes of $V = V_d \cdots V_1$ should be within δ of the “correct” probabilities of the different outcomes of $U = U_d \cdots U_1$, i.e. $|p_U - p_V| \leq \delta$.

How small must ε be with respect to δ in order for us to achieve this?

Hint: recall that $|p_U - p_V| \leq 2\|U - V\|$.

12.11.7 Statistical distance and a special event

1. Show that, if p and q are probability distributions on the same sample space Ω , then

$$d(p, q) = \max_{A \subseteq \Omega} \{|p(A) - q(A)|\}.$$

2. By definition, the above maximum is realised for some specific subset $A \subseteq \Omega$, i.e. there exists some event (described by the set of outcomes A) that is optimal in distinguishing p from q . What is this event?

12.11.8 Joint probability distributions

If we simultaneously sample two random variables from the same probability space, then we obtain a **joint distribution**:

$$r(x, y) := \Pr(x \text{ and } y).$$

From this we can recover the **marginals**

$$\begin{aligned} p(x) &:= \sum_y r(x, y) \\ q(y) &:= \sum_x r(x, y). \end{aligned}$$

So let $r(x, y)$ be a joint probability distribution with marginals $p(x)$ and $q(y)$. Show that

$$d_{\text{tr}}(p, q) \leq \Pr(x \neq y) \equiv \sum_{\{x, y | x \neq y\}} p(x, y).$$

Hint:

$$\begin{aligned} d_{\text{tr}}(p, q) &= 1 - \sum_x \min\{p(x), q(x)\} \\ &\leq 1 - \sum_x p(x, x) \\ &= \Pr(x \neq y). \end{aligned}$$

12.11.9 Distinguishability and the trace distance

Say we have a physical system which is been prepared in one of two states (say, ρ_0 and ρ_1), each with equal probability. Then, as shown in Section 12.8, a single measurement can distinguish between the two preparations with probability at most $\frac{1}{2}[1 + d_{\text{tr}}(\rho_0, \rho_1)]$.

1. How does this probability change if the states ρ_0 and ρ_1 are *not* equally likely, but instead sent with some predetermined probabilities p_0 and p_1 , respectively?
2. Suppose that you are given one randomly selected qubit from a pair in the state

$$|\psi\rangle = \frac{1}{\sqrt{2}} \left(|0\rangle \otimes \left(\sqrt{\frac{2}{3}}|0\rangle - \sqrt{\frac{1}{3}}|1\rangle \right) + |1\rangle \otimes \left(\sqrt{\frac{2}{3}}|0\rangle + \sqrt{\frac{1}{3}}|1\rangle \right) \right)$$

from Exercise 8.8.1. What is the maximal probability with which we can determine which qubit (either the first or the second) we were given?

13 Decoherence and recoherence

*About the one big problem that hinders us from physically implementing everything that we've learnt so far: **decoherence**. But also about how we can start to deal with it via some elementary **error correction**, including the **Shor** [[9, 1, 3]] **quantum code**, which generalises the classical **three-bit repetition code**.*

As the adage goes, “in theory, theory and practice rarely differ; in practice, they often do”. *In theory*, we know how to build a quantum computer: we can start with simple quantum logic gates and try to integrate them together into quantum networks. However, if we keep on putting quantum gates together into networks we will quickly run into some serious problems *in practice*: the more interacting qubits involved, the harder it is to prevent them from getting entangled with the environment. This unwelcome entanglement, also known as **decoherence**, destroys the interference, and thus the power, of quantum computing. To counteract this problem, we will start to look at the idea of **error correcting codes**, which protect our data against unwanted errors, but at the cost of encoding it across more ancillary qubits.

13.1 The three-qubit code

In Section 9.3 we met the notion of isometries: operators V that map one Hilbert space to another and satisfy $V^\dagger V = \mathbf{1}$. This implies that isometries can be reversed, or **corrected**: we can apply V^\dagger and end up exactly how we started.

We say that a quantum channel $\mathcal{E}: \mathcal{B}(\mathcal{H}) \rightarrow \mathcal{B}(\mathcal{H}')$ is **correctable** if there exists a **recovery channel** $\mathcal{R}: \mathcal{B}(\mathcal{H}') \rightarrow \mathcal{B}(\mathcal{H})$ such that the composition $\mathcal{R} \circ \mathcal{E}$ is the identity channel $\mathbf{1}$.

Now suppose we have isometries $V_1, \dots, V_n: \mathcal{H} \rightarrow \mathcal{H}'$. If \mathcal{H}' is “sufficiently bigger” than \mathcal{H} , and if the images $\mathcal{H}'_i := V_i(\mathcal{H})$ do not overlap then we can reverse the action of the channel given by a statistical mixture of the V_i : we can, at least in principle, perform a measurement on \mathcal{H}' , defined by the partition $\mathcal{H}' = \mathcal{H}'_1 \oplus \mathcal{H}'_2 \oplus \dots \oplus \mathcal{H}'_n$, and find out which subspace contains the output state; once we know which subspace the input was sent to, we know which particular isometry V_k was applied by the channel; then we simply apply V_k^\dagger .

More precisely, we say that the \mathcal{H}'_i “do not overlap” to mean that the subspaces \mathcal{H}'_i are mutually orthogonal

Apart from individual unitaries or isometries, the only **correctable** channels are exactly the statistical mixtures of $\{V_i\}$ such that $V_i^\dagger V_j = \delta_{ij} \mathbf{1}$, i.e. mixtures of mutually orthogonal isometries.

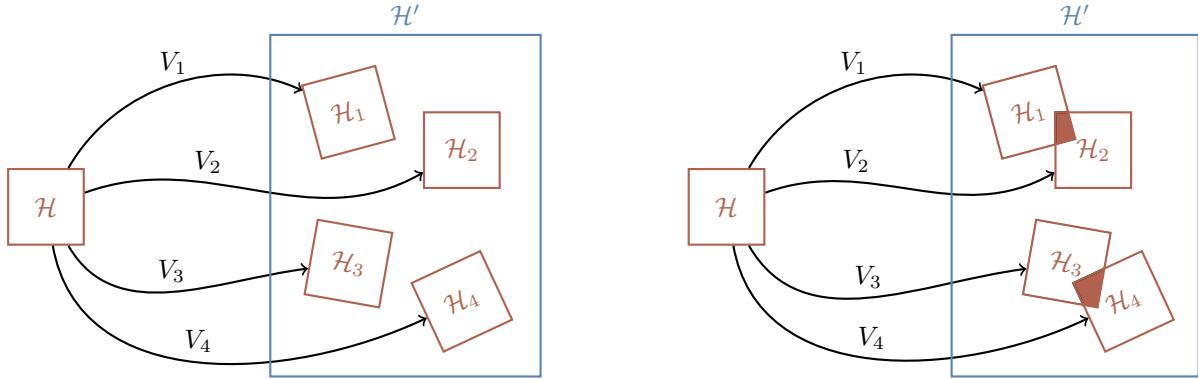


Figure 13.1: A visualisation of correctable (left) and non-correctable (right) channels. Each isometry V_i , which is chosen with some probability p_i , maps the original space to a different space. If those spaces do not overlap, we can detect which one we're in and hence compensate (i.e. correct). If the two spaces partially coincide, however, then there exist states for which we *cannot* detect which isometry occurred.

Here is a simple but important example: the **three-qubit code**. Take a qubit in some pure state $|\psi\rangle = \alpha|0\rangle + \beta|1\rangle$, introduce two auxiliary qubits in a fixed state $|0\rangle|0\rangle$, and apply a unitary operation to the three qubits, namely two controlled-NOT gates:

$$\begin{array}{c} \alpha|0\rangle + \beta|1\rangle \\ |0\rangle \\ |0\rangle \end{array} \xrightarrow{\quad \begin{array}{c} \text{---} \\ \oplus \\ \text{---} \end{array}} \left. \begin{array}{c} \text{---} \\ \oplus \\ \text{---} \end{array} \right\} \alpha|000\rangle + \beta|111\rangle$$

The result is the isometric embedding of the 2-dimensional Hilbert space of the first qubit (spanned by $|0\rangle$ and $|1\rangle$) into the 2-dimensional subspace (spanned by $|000\rangle$ and $|111\rangle$) of the 8-dimensional Hilbert space of the three qubits. The isometric operator

$$V = |000\rangle\langle 0| + |111\rangle\langle 1|$$

acts via

$$\alpha|0\rangle + \beta|1\rangle \mapsto \alpha|000\rangle + \beta|111\rangle.$$

This three qubit-encoding can be reversed by the mirror image circuit:

$$\begin{array}{c} \alpha|0\rangle + \beta|1\rangle \\ |0\rangle \\ |0\rangle \end{array} \xleftarrow{\quad \begin{array}{c} \text{---} \\ \oplus \\ \text{---} \end{array}} \left. \begin{array}{c} \text{---} \\ \oplus \\ \text{---} \end{array} \right\} \alpha|000\rangle + \beta|111\rangle$$

This isometry is just one member of a family, and we will spend the rest of this chapter building up to the general theory, and understanding how this three-qubit encoding is useful in error correction.

Let's start with the following scenario. Alice constructs a quantum channel which is a mixture of four isometries. The input is a single qubit, and the output is a dilated system composed of three qubits. She prepares the input qubit in a state $|\psi\rangle$ and then combines it with the two ancillary qubits which are in a fixed state $|0\rangle|0\rangle$. Then she

We will return to this example, using the language of stabilisers from Chapter 7, in Chapter 14.

Our arguments here can be easily extended to any mixed state ρ , but for simplicity we consider the case of a pure state.

applies one of the four, randomly chosen, unitary operations to the three qubits, to generate the following four isometries:

$$V_{00} = |000\rangle\langle 0| + |111\rangle\langle 1|$$

$$V_{01} = |001\rangle\langle 0| + |110\rangle\langle 1|$$

$$V_{10} = |010\rangle\langle 0| + |101\rangle\langle 1|$$

$$V_{11} = |100\rangle\langle 0| + |011\rangle\langle 1|.$$

The three qubits, which form the output of the channel, are given to Bob, whose task is to recover the original state $|\psi\rangle$ of the input qubit. In this scenario, Bob, who knows the four isometries, can find out which particular isometry was applied. He knows that

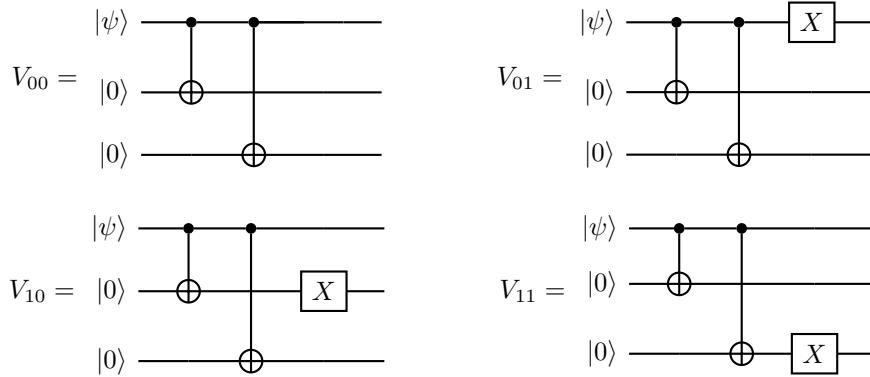
- V_{00} maps \mathcal{H} to \mathcal{H}'_{00} , which is a subspace of \mathcal{H}' spanned by $|000\rangle$ and $|111\rangle$;
- V_{01} maps \mathcal{H} to \mathcal{H}'_{01} , which is a subspace of \mathcal{H}' spanned by $|001\rangle$ and $|110\rangle$;
- V_{10} maps \mathcal{H} to \mathcal{H}'_{10} , which is a subspace of \mathcal{H}' spanned by $|010\rangle$ and $|101\rangle$;
- V_{11} maps \mathcal{H} to \mathcal{H}'_{11} , which is a subspace of \mathcal{H}' spanned by $|100\rangle$ and $|011\rangle$.

Given that these subspaces are mutually orthogonal, and $\mathcal{H}' = \mathcal{H}'_{00} \oplus \mathcal{H}'_{01} \oplus \mathcal{H}'_{10} \oplus \mathcal{H}'_{11}$, Bob can perform a measurement defined by the projectors on these subspaces. For example, if Alice randomly picked V_{01} , then the input state $|\psi\rangle = \alpha|0\rangle + \beta|1\rangle$ will be mapped to the output state $\alpha|001\rangle + \beta|110\rangle$ in the \mathcal{H}'_{01} subspace. Bob's measurement

$$P_{01} = |001\rangle\langle 001| + |101\rangle\langle 101|$$

will then detect \mathcal{H}'_{01} as the subspace where the output state resides, but the measurement (i.e. the corresponding projection) will not affect any state in that subspace. Bob can now simply apply V_{01}^\dagger and obtain $|\psi\rangle$.

Below is a diagram of how the four isometries are implemented. We will see how to reverse these operations in Section 13.2.



13.2 Towards error correction

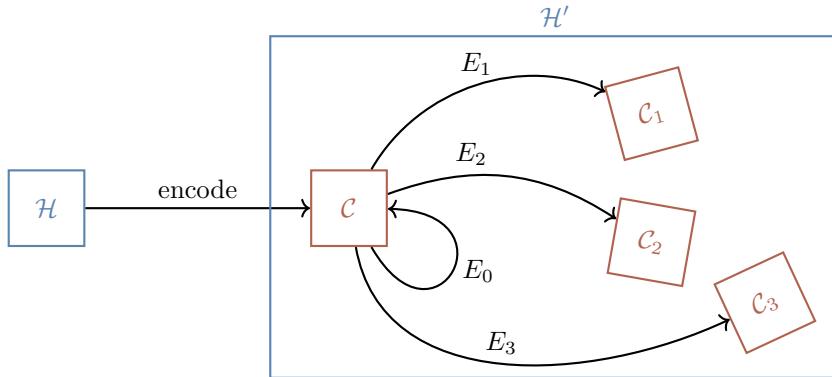
In Section 13.1, when Alice used a random choice of four isometries to produce a three-qubit output, notice how we can write

$$V_{01} = (\mathbf{1} \otimes \mathbf{1} \otimes X)V_{00}$$

$$V_{01} = (\mathbf{1} \otimes X \otimes \mathbf{1})V_{00}$$

$$V_{01} = (X \otimes \mathbf{1} \otimes \mathbf{1})V_{00}$$

and thus express all of the isometries in terms of V_{00} . In other words, rather than thinking of Alice as picking randomly between four different isometries, we can imagine that she *always* picks the **encoding** isometry V_{00} , and then some noisy process randomly applies one of the four actions $\mathbf{1}$, $\mathbf{11}X$, $\mathbf{1}X\mathbf{1}$, or $X\mathbf{11}$.



Correcting the isometry then corresponds to identifying which error happened, fixing it, and then removing the encoding: a process known as **decoding**. This is the format of **error correction** in a nutshell.

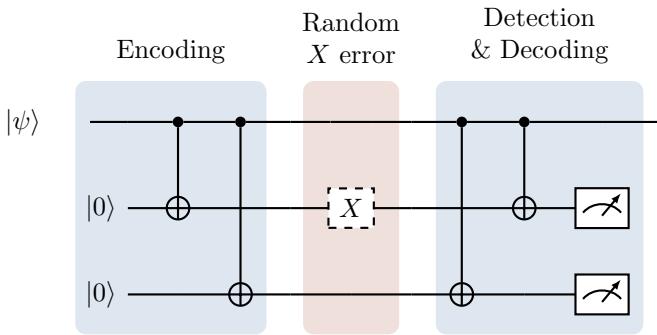


Figure 13.2: Quantum error correction can be thought of as a three-step process: encoding, transmitting through a noisy channel, and then detecting and decoding. We will give a more accurate depiction of an error correcting diagram, explaining what actually happens in the “detection & decoding portion” in Figure 13.7.

When we studied Pauli stabilisers in Section 7.2, we came across exactly the spaces of this example:

$+1$	$ZZ1$	-1
$ 000\rangle$ $ 111\rangle$		$ 100\rangle$ $ 011\rangle$
$++$		$-+$

$+1$	$1ZZ$	-1
$+-$		$--$
$ 001\rangle$ $ 110\rangle$		$ 010\rangle$ $ 101\rangle$

The stabiliser formalism gives us a very natural way of describing the error correcting code, along with its correction:

- the **codespace** (i.e. the space with no error) is defined by the two stabilisers $+ZZ1$ and $+1ZZ$
- the error can be determined by measuring the value of these two stabilisers; we simply have to find which of the four possible errors gives the correct (anti-)commutation relations as specified by the measurement outcomes ± 1 .

Generalising this idea further is the subject of Section 13.6.

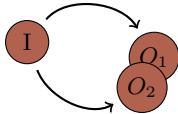
If a set $\{V_x\}$ of correctable isometries are related by

$$V_x = U_x V_0$$

for some set of unitaries $\{U_x\}$ with $U_0 = \mathbf{1}$, then an **encoding** operation V_0 provides protection against the **errors** U_x .

13.3 Discretisation of quantum errors

When a quantum computer interacts and becomes entangled with its environment, it impacts the environment in such a way that the environment maintains a *physical record* of how the computer arrived at the desired output. Here, in our simplistic diagram, we consider only two computational paths.



With decoherence present, quantum computation spills out the environment and results in not one, but two output states:

$$\begin{aligned} |O_1\rangle &:= |O\rangle|e_1\rangle \\ &= \text{"computer shows output } O, \text{ environment knows that path 1 was taken"} \\ |O_2\rangle &:= |O\rangle|e_2\rangle \\ &= \text{"computer shows output } O, \text{ environment knows that path 2 was taken"} \end{aligned}$$

The two final states O_1 and O_2 are identical if and only if $|\langle e_1 | e_2 \rangle| = 1$. In this case, the environment does not know anything about what happened during the computation — there is quantum interference — and we add probability amplitudes corresponding to the two computational paths. In contrast, the two final states O_1 and O_2 are completely different if and only if $|\langle e_1 | e_2 \rangle| = 0$. Then there is only one path to the output — there is no quantum interference — and there is nothing to add. Of course, there are also midway cases $0 < |\langle e_1 | e_2 \rangle| < 1$ corresponding to partial distinguishability of the final states.

If we wish to study the evolution of the qubit alone, then we can do so in terms of density operators: it evolves from the pure state $|\psi\rangle\langle\psi|$ to a mixed state, which can be obtained by tracing over the environment. We know that the state vector $|\psi\rangle = \alpha|0\rangle + \beta|1\rangle$ evolves as

$$(\alpha|0\rangle + \beta|1\rangle)|e\rangle \mapsto \alpha|0\rangle|e_{00}\rangle + \beta|1\rangle|e_{11}\rangle$$

and we can write this as the evolution of the projector $|\psi\rangle\langle\psi|$, and then trace over the environment to obtain

$$|\psi\rangle\langle\psi| \mapsto |\alpha|^2|0\rangle\langle 0|e_{00}\rangle + \alpha\beta^*|0\rangle\langle 1|e_{11}\rangle + \alpha^*\beta|1\rangle\langle 0|e_{00}\rangle + |\beta|^2|1\rangle\langle 1|e_{11}\rangle.$$

Written in matrix form, this is

$$\begin{bmatrix} |\alpha|^2 & \alpha\beta^* \\ \alpha^*\beta & |\beta|^2 \end{bmatrix} \mapsto \begin{bmatrix} |\alpha|^2 & \alpha\beta^*\langle e_{11}|e_{00}\rangle \\ \alpha^*\beta\langle e_{00}|e_{11}\rangle & |\beta|^2 \end{bmatrix}.$$

The off-diagonal elements (originally called **coherences**) vanish as $\langle e_{00}|e_{11}\rangle$ approaches zero. This is why this particular interaction is called **decoherence**.

Notice that

$$|\psi\rangle|e\rangle \mapsto \mathbf{1}|\psi\rangle|e_1\rangle + Z|\psi\rangle|e_Z\rangle,$$

implies

$$|\psi\rangle\langle\psi| \mapsto \mathbf{1}|\psi\rangle\langle\psi|\mathbf{1}\langle e_1|e_1\rangle + Z|\psi\rangle\langle\psi|Z\langle e_Z|e_Z\rangle,$$

only if $\langle e_1|e_Z\rangle = 0$, since otherwise we would have additional cross terms $\mathbf{1}|\psi\rangle\langle\psi|Z$ and $Z|\psi\rangle\langle\psi|\mathbf{1}$. In this case (i.e. when $\langle e_1|e_Z\rangle = 0$) we can indeed say that, with probability $\langle e_1|e_1\rangle$, nothing happens, and, with probability $\langle e_Z|e_Z\rangle$, the qubit undergoes the phase-flip Z . We can also represent this with the Kraus operators

$$E_0 = \sqrt{\langle e_1|e_1\rangle}\mathbf{1}$$

$$E_1 = \sqrt{\langle e_Z|e_Z\rangle}Z$$

which can be shown to satisfy $E_0^\dagger E_0 + E_1^\dagger E_1 = \mathbf{1}$.

The process of decoherence is continuous. It involves the environment gradually acquiring information about computational paths and the associated relative environmental states ($|e_0\rangle$ and $|e_1\rangle$ in our example above), which evolve over time to become increasingly orthogonal to one another. Despite this, we can perceive the influence of the environment on our system of interest — a collection of qubits in a quantum computer — in terms of discrete operations. In essence, *we can digitise quantum errors*.

13.4 Digitising quantum errors

The most general qubit-environment interaction is of the form

$$|0\rangle|e\rangle \mapsto |0\rangle|e_{00}\rangle + |1\rangle|e_{01}\rangle$$

$$|1\rangle|e\rangle \mapsto |1\rangle|e_{10}\rangle + |0\rangle|e_{11}\rangle$$

where the states of the environment are neither normalised nor orthogonal. This leads to decoherence

$$\begin{aligned} (\alpha|0\rangle + \beta|1\rangle)|e\rangle &\mapsto (\alpha|0\rangle + \beta|1\rangle)\frac{|e_{00}\rangle + |e_{11}\rangle}{2} \\ &\quad + (\alpha|0\rangle - \beta|1\rangle)\frac{|e_{00}\rangle - |e_{11}\rangle}{2} \\ &\quad + (\alpha|1\rangle + \beta|0\rangle)\frac{|e_{01}\rangle + |e_{10}\rangle}{2} \\ &\quad + (\alpha|1\rangle - \beta|0\rangle)\frac{|e_{01}\rangle - |e_{10}\rangle}{2}. \end{aligned}$$

which can be written as

$$|\psi\rangle|e\rangle \mapsto \mathbf{1}|\psi\rangle|e_1\rangle + Z|\psi\rangle|e_Z\rangle + X|\psi\rangle|e_X\rangle + Y|\psi\rangle|e_Y\rangle.$$

The intuition behind this expression is that four things can happen to the qubit:

1. nothing (1)
2. phase-flip (Z)
3. bit-flip (X)
4. both bit-flip and phase-flip (Y).

This is certainly the case when the states $|e_1\rangle, |e_X\rangle, |e_Y\rangle$ and $|e_Z\rangle$ are mutually orthogonal, but if this is not so then we cannot perfectly distinguish between the four alternatives.

We can reduce quantum errors in this general scenario to just *two types*: bit-flip errors X , and phase-flip errors Z .

In short, *if we can correct Pauli errors then we can correct all errors.*

We will soon stop warning that this intuition is not entirely accurate, so keep it in mind!

In general, given n qubits in state $|\psi\rangle$, and an environment in state $|e\rangle$, the joint evolution can be expanded as

$$|\psi\rangle|e\rangle \mapsto \sum_{i=1}^{4^n} E_i |\psi\rangle|e_i\rangle,$$

The sum is from $i = 1$ to 4^n because there are 4^n different (tensor products of) Pauli operators acting on n qubits.

where the E_i are the n -fold tensor products of the Pauli operators and the $|e_i\rangle$ are the corresponding states of the environment (which, again, are not assumed to be normalised or mutually orthogonal). For example, in the case $n = 5$, a typical operator E_i may look like

$$X \otimes Z \otimes \mathbf{1} \otimes \mathbf{1} \otimes Y \equiv XZ\mathbf{1}\mathbf{1}Y.$$

We say that such an E_i represents an error consisting of the bit error (or X error) on the first qubit, phase error (or Z error) on the second qubit, and both bit and phase error (or Y error) on the fifth qubit.

In terms of density operators, we have a quantum channel described by the Kraus operators E_i above

$$\rho \mapsto \sum_i E_i \rho E_i^\dagger$$

One final time: *this is not entirely accurate if the corresponding states of the environment are not mutually orthogonal*, but it gives the right kind of intuition nonetheless.

that acts on any input state ρ , be it mixed or pure. In particular, this channel turns the pure state $|\psi\rangle$ into a statistical mixture of states $|\tilde{\psi}_i\rangle = E_i|\psi\rangle$. Note that the $|\tilde{\psi}_i\rangle$ are not normalised:

$$p_i := \langle \tilde{\psi}_i | \tilde{\psi}_i \rangle = \langle \psi | E_i^\dagger E_i | \psi \rangle$$

is exactly the probability with which the normalised version of $E_i|\psi\rangle$ appears in the mixture. This mixture may arise if one measures the environment in the $|e_i\rangle$ basis and then forgets about the result.

13.5 Recoherence

If we could measure the environment in the $|e_i\rangle$ basis without forgetting the result, then we would know what kind of error had occurred (say, some E_k) and could then simply restore the original state $|\psi\rangle$ by reversing the action of E_k . But the challenge lies in our lack of control over the environment. At first glance, once our bunch of qubits, initially in state $|\psi\rangle$, gets entangled with the environment

$$|\psi\rangle|e\rangle \mapsto \sum_i E_i |\psi\rangle|e_i\rangle$$

the situation looks rather hopeless — we do not have any control over the environment. However, there is a way around this. We can couple the qubits to an

auxiliary system that we *do* control (an ancilla), and then attempt to transfer the qubits–environment entanglement to a qubits–ancilla entanglement. In other words, we prepare the ancilla in some prescribed state $|a\rangle$ and try to undo the decoherence \mathcal{E} using some **recovery** or **recoherence** operator \mathcal{R} that acts as

$$|\psi\rangle|a\rangle \mapsto \sum_k R_k |\psi\rangle|a_k\rangle.$$

Thus decoherence followed by recoherence acts as

$$\begin{aligned} \mathcal{RE}: |\psi\rangle|e\rangle|a\rangle &\mapsto \sum_i E_i |\psi\rangle|e_i\rangle|a\rangle \\ &\mapsto \sum_{i,k} R_k E_i |\psi\rangle|e_i\rangle|a_k\rangle \end{aligned}$$

which we can also express in a diagram, as in Figure 13.3.

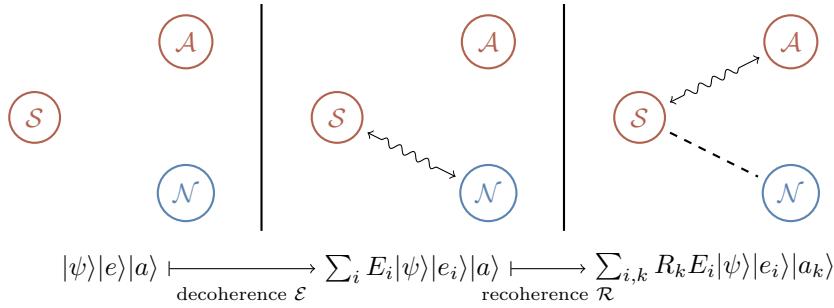


Figure 13.3: The initial state undergoing decoherence followed by recoherence. Here S is the system of qubits that we want to work with, N is the environment, and A is the ancilla that we introduce. The squiggly arrow represents interactions, and the dashed line represents entanglement.

But for this to help us, we need to end up with a state where the ancilla and the environment are entangled with one another, and the qubit is entangled with nothing, i.e. a state of the form

$$|\psi\rangle \otimes (\text{some entangled state of the ancilla and environment})$$

as shown in Figure 13.4

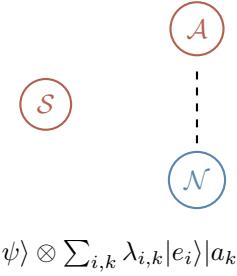


Figure 13.4: The desired outcome of the decoherence–recoherence process.

Ideally we would like this to hold for *all* states $|\psi\rangle$, but this turns out to be too much to ask: as we shall see in a moment, we will have to confine our recoverable states to those that belong to a subspace called the **codespace**. But then at least for these states we expect to have

$$\sum_{i,k} R_k E_i |\psi\rangle|e_i\rangle|a_k\rangle = \sum_{i,k} |\psi\rangle \otimes \lambda_{ik} |e_i\rangle|a_k\rangle.$$

The ability of R_k to perform the correction like this means that

$$R_k E_i = \lambda_{ik} \mathbf{1}$$

when acting on the codespace states $|\psi\rangle$. In turn, this means that

$$\begin{aligned} \sum_k (R_k E_j)^\dagger (R_k E_i) &= E_j^\dagger \left(\sum_k R_k^\dagger R_k \right) E_i \\ &= E_j^\dagger E_i \\ &= \lambda_{jk}^* \lambda_{ik} \mathbf{1} \end{aligned}$$

which reminds us of the hopefully now-familiar condition for being able to correct a randomly chosen isometry.

Last but not least, note that the decoherence operator \mathcal{R} not only allows us to recover from the errors E_i , but also from any errors that are in the linear span of these. Thus if the errors E_i form a basis in the matrix space — as is the case for the Pauli matrices together with the identity — then once we design an error recovery scheme for the E_i , we will be able to correct *any* error.

If a quantum error correction method corrects errors E_1 and E_2 , then it also corrects any linear combination of E_1 and E_2 .

For example, in Section 13.2 we saw how the three-qubit encoding could correct for one of the possible errors $\mathbf{1}$, X_1 , X_2 , or X_3 . But not only can we correct for this discrete set of errors, we can also correct for any linear combination of them, such as $R_{X_1}(\theta)$, which acts as

$$|\psi\rangle \xrightarrow{R_{X_1}(\theta)} \cos \frac{\theta}{2} |\psi\rangle + i \sin \frac{\theta}{2} X_1 |\psi\rangle.$$

In other words, the scenario where, with probability $\cos^2 \frac{\theta}{2}$ we find that no error occurred, and with probability $\sin^2 \frac{\theta}{2}$ we find that the error X_1 occurred and correct it.

13.6 The classical repetition code

We have now essentially met all the concepts of quantum error correction, but everything has been phrased in terms of correctable isometries. Now we need to repeat much the same information from a slightly different perspective, including a brief detour to introduce some concepts and methods from the classical theory of error correction. Even though quantum problems often require novel solutions, it is always a good idea to look at the classical world to see if there is anything equivalent there, and, if so, how people deal with it. In this particular case, once we have digitised quantum errors, we can see that quantum decoherence is a bit like classical noise (i.e. bit-flips), except that we have two types of errors: these classical bit-flips, and then the purely quantum phase-flips. But there's one key thing that we know relating these two types of errors, namely that phase-flips can be turned into bit-flips by sandwiching them between Hadamard transforms:

$$HZH = X.$$

This opens up the possibility of adopting classical techniques of error correction to the quantum case. There is a vast body of knowledge out there about classical error-correcting codes, and we can only scratch the surface here.

Quite often we will write X_i to mean “an X error on the i -th qubit”, so that e.g. $X_2 = \mathbf{1} \otimes X_2 \otimes \mathbf{1} \dots \otimes \mathbf{1}$.

Suppose you want to send a k -bit message across a noisy channel. If you choose to send the message directly, some bits may be flipped, and a different message will likely arrive with a certain number of errors. Let's suppose that the transmission channel flips each bit in transit with some fixed probability p . If this error rate is considered too high, then it can be decreased by introducing some redundancy. For example, we could encode each bit into three bits:

$$\begin{aligned} 0 &\mapsto 000 \\ 1 &\mapsto 111. \end{aligned}$$

These two binary strings, 000 and 111, are called **codewords**. Beforehand, Alice and Bob agree that, from now on, each time you want to send a **logical 0**, you will encode it as 000, and each time you want to send a **logical 1**, you will encode it as 111.

Here's an example. Say that Alice wants to send the message 1011. She first encodes it as the string

$$111\ 000\ 111\ 111$$

and then sends it to Bob over the noisy channel. Note that she is now not sending just four, but instead *twelve* physical bits. This is more costly (in terms of time or energy, or maybe even money), but might be worth it to ensure a more reliable transmission. Let's say that Bob then receives the message

$$110\ 010\ 101\ 100.$$

Clearly some errors have occurred! In fact, even Bob knows this, because he expects to receive 3-bit chunks of either all 0s or all 1s. He uses the “majority vote” decoding method:

- 110 is decoded as 1
- 010 is decoded as 0
- 101 is decoded as 1
- 100 is decoded as 0.

As we can see, if a triplet contains either zero or one errors then the decoding returns the correct bit value, otherwise it errs. In our example, the first three triplets are correctly decoded, but the fourth suffered two errors and is thus wrongly decoded as 0. This whole process can be represented as

$$1011 \xrightarrow{\text{encoding}} 111\ 000\ 111\ 111 \xrightarrow{\text{noise}} 110\ 010\ 101\ 100 \xrightarrow{\text{decoding}} 1010.$$

The noisy channel flipped 5 out of the 12 physical bits, and the whole encoding-decoding process reduced this down to only one logical error.

We can make a simple estimate on how good this scheme will be. Assuming that the errors are independent then, for any given triplet,

$$\begin{cases} \text{no errors} & \text{probability } (1-p)^3 \\ 1 \text{ error} & \text{probability } 3p(1-p)^2 \\ 2 \text{ errors} & \text{probability } 3p^2(1-p) \\ 3 \text{ errors} & \text{probability } p^3. \end{cases}$$

The assumption that the errors are independent is very important, but not always physically realistic! We will re-examine this assumption a few times in Chapter 14.

More succinctly, the probability that n errors occur is

$$\binom{3}{n} p^n (1-p)^{3-n}$$

where $\binom{m}{n} = m!/(n!(m-n)!)$ is the **binomial coefficient**. Given that the scheme decodes correctly exactly when we have at most one error, the net probability of errors is just the probability that either two or three errors occur, which is

$$3p^2(1-p) + p^3.$$

This means that our encoding-decoding scheme actually lowers the probability of error if and only if

$$3p^2(1-p) + p^3 < \frac{1}{2}$$

i.e. when $p < 1/2$. This important number is known as the **error threshold** of the code, and is a good judge of how useful a code actually is. When p is really small, we can basically ignore the p^3 term, since it is even smaller still, and claim that the error probability is reduced from p to roughly $3p^2$.

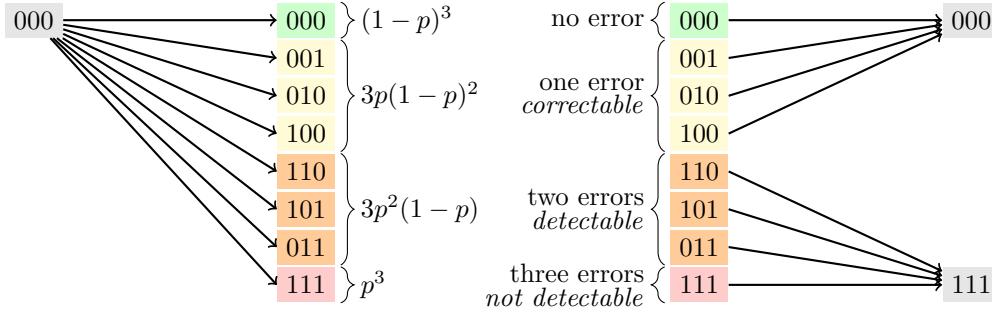


Figure 13.5: What can happen, and the respective probabilities, when we transmit the codeword 000. With this scheme, we can correct up to one error, and detect up to two. Note that when two errors occur, the “majority vote” correction scheme actually gives the *wrong* “correction”.

In this example, we can see that if one or two errors occur, then the resulting bit string will no longer be a valid codeword, which makes the error detectable. For example, if the bit string 101 appears in our message, then we know that some error must have occurred, because 101 is not a codeword, so we never would have sent it. We can't be certain *which* error has occurred, since it could have been a single bit-flip on 111 (more likely) or a double bit-flip on 000 (less likely), but we know for sure that *one of these two* went wrong. In the worst case scenario, where three bit-flips happen, the error will be undetectable: it will turn 000 into 111 (and vice versa). This results in what is known as a **logical error**, where the corrupted string is also a codeword, leaving us with no indication that anything has gone wrong.

Below a certain error threshold, the three-bit code improves the reliability of the information transfer. This simple repetition code encoded one bit into three bits, and corrected up to one error. In general, there are classical codes that can encode k bits into n bits and correct up to r errors. One more important number that we need is the **distance** of such an encoding, which is defined to be the minimum number of errors that can pass undetected (or, equivalently, the minimum Hamming distance between two codewords). More generally, the distance d relates to the number of errors t that a code can correct by $d = 2t + 1$, or, equivalently, $t = \lfloor d/2 \rfloor$.

Looking back again at Figure 13.5, we see that if exactly one or two errors occur in our three-bit code then we can detect that an error has occurred, since we will have a string where not all of the bits are the same, which means that it is definitely not one of our code words. However, if three errors occur then the errors are not only impossible to correct, but they are also *impossible to detect*. So the code that we have described has $n = 3$, $k = 1$, and $d = 3$.

That is, the number of bit-flips required to move from one to the other; recall Section 12.1.

A code that encodes k bits into n bits and has a distance of d is called an $[n, k, d]$ **code**. The **rate** of an $[n, k, d]$ code is defined to be $R = k/n$.

In an $[n, k, d]$ code, the encoder divides the message into chunks of k bits and

encodes each k -bit string into a pre-determined n -bit codeword. There are 2^k distinct codewords among all 2^n binary strings of length n . The recipient then applies the decoder, which takes chunks of n bits, looks for the nearest codeword (in terms of Hamming distance), and then decodes the n -bit string into that k -bit codeword. For example, in our 3-bit repetition code, we have the two codewords 000 and 111, among all eight binary strings of length 3, as shown in Figure 13.6.

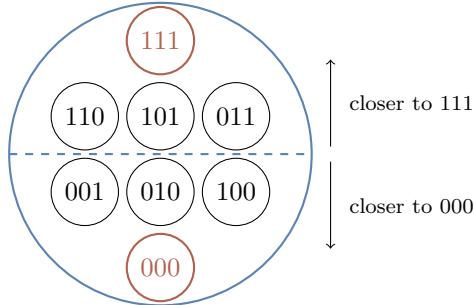


Figure 13.6: All the 3-bit strings that are within Hamming distance 1 from 000 are below the line, and all those that are within Hamming distance 1 from 111 are above the line. The decoder assumes that the former are corrupted versions of 000, and the latter of 111.

13.7 Correcting bit-flips

In order to protect a qubit against bit-flips (thought of as incoherent X rotations), we rely on the same classical repetition code as in Section 13.6, but both encoding and error correction are now implemented by quantum operations. Let's return to the example of the three-qubit code that we introduced in Section 13.1. We take a qubit in some unknown pure state $\alpha|0\rangle + \beta|1\rangle$ and encode it into three qubits, introducing two auxiliary qubits:

$$\begin{array}{c} \alpha|0\rangle + \beta|1\rangle \\ |0\rangle \\ |0\rangle \end{array} \xrightarrow{\quad \begin{array}{c} \text{CNOT} \\ \oplus \\ \text{CNOT} \end{array}} \left. \begin{array}{c} \alpha|000\rangle + \beta|111\rangle \\ \end{array} \right\}$$

Mathematically, this is an isometric embedding of a two-dimensional space into an eight-dimensional one. It is important to note that this is *not* just the classical repetition code “but with qubits instead of bits” — this would be *impossible* to construct, since the no-cloning theorem tells us that we can never build a circuit that enacts

$$\alpha|0\rangle + \beta|1\rangle \mapsto (\alpha|0\rangle + \beta|1\rangle)(\alpha|0\rangle + \beta|1\rangle)(\alpha|0\rangle + \beta|1\rangle).$$

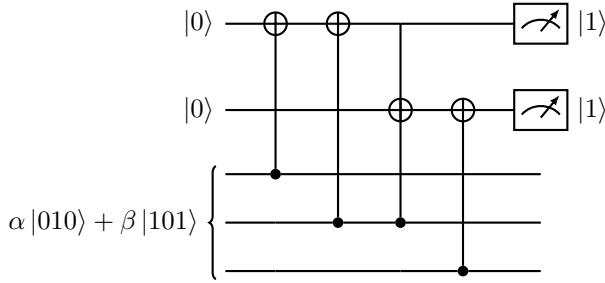
Rather than repeating the qubit like this, the three-qubit code sort of “smears it out” across three qubits, resulting in the *entangled* state $\alpha|000\rangle + \beta|111\rangle$.

Now suppose that one qubit is flipped, say, the second one. The encoded state then becomes $\alpha|010\rangle + \beta|101\rangle$. Decoding requires some care: measuring the three qubits directly would destroy the superposition that we are working so hard to protect. So instead we introduce two ancilla qubits, both in state $|0\rangle$, and apply the following circuit:

All the codes we will study have encoding circuits that can be constructed out of controlled-NOT and Hadamard gates: we are dealing with *Clifford circuits* (Section 7.7).

Recall that an isometry is the generalisation of a unitary but where we are also allowed to bring in additional qubits.

There is a subtle difference between **decoding** and **unencoding**: the latter consists of simply reversing the encoding process; the former consists of using the results of measurements (the **error syndrome**) to perform a more adapted “unencoding”.



This decoding circuit is exactly the same as the ones for measuring the Pauli stabilisers $ZZ1$ and $1ZZ$ (as described in Section 7.4).

Measuring the two ancilla qubits gives us what is known as the **error syndrome** (or sometimes just **syndrome**, for short), which tells us how to correct the three qubits (known as the **data qubits**) of the code. The theory behind this works as follows:

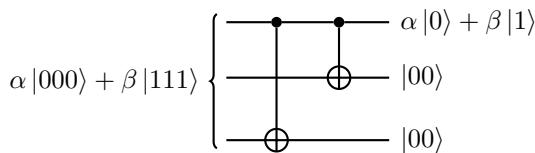
- if the first and second (counting from the top) data qubits are in the same state then the first ancilla will be in the $|0\rangle$ state; otherwise the first ancilla will be in the $|1\rangle$ state
- if the second and third data qubits are in the same state then the second ancilla will be in the $|0\rangle$ state; otherwise the second ancilla will be in the $|1\rangle$ state.

So the four possible error syndromes each indicate a different scenario:

- $|00\rangle$: no error
- $|01\rangle$: bit-flip in the first data qubit
- $|10\rangle$: bit-flip in the second data qubit
- $|11\rangle$: bit-flip in the third data qubit.

Again, for now we are assuming that *at most one* bit-flip error occurs.

In our example, the error syndrome is $|11\rangle$, and so we know that the first and second qubits differ, as do the second and third. This means that the first and third must be the same, and the second suffered the bit-flip error. Knowing the error, we can now fix it by applying an X gate to the second qubit. The final result is the state $\alpha|000\rangle + \beta|111\rangle$, which is then turned into $(\alpha|0\rangle + \beta|1\rangle)|00\rangle$ by running the mirror image of the encoding circuit:



It is also important to note that the actual error correction can be implemented by a single unitary operation U_c on the five total qubits, with

$$\begin{aligned} U_c = & (|0\rangle\langle 0| \otimes |0\rangle\langle 0|) \otimes (\mathbf{111}) \\ & + (|0\rangle\langle 0| \otimes |1\rangle\langle 1|) \otimes (\mathbf{11}X) \\ & + (|1\rangle\langle 1| \otimes |0\rangle\langle 0|) \otimes (X\mathbf{11}) \\ & + (|1\rangle\langle 1| \otimes |1\rangle\langle 1|) \otimes (\mathbf{1}X\mathbf{1}). \end{aligned}$$

We draw the general circuit for bit-flip protection in Figure ??, writing out this U_c in full, denoting the error-syndrome measurement by a, b .

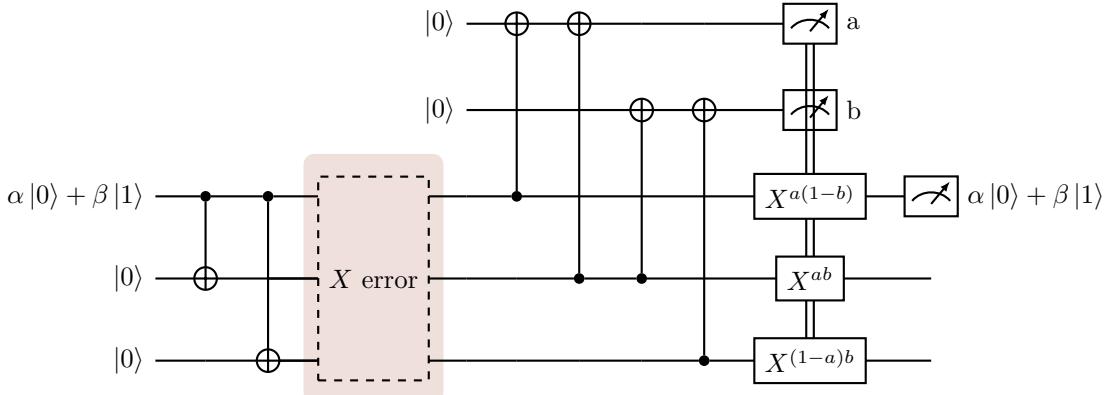


Figure 13.7: The quantised version of the classical [3, 1, 3] code. If at most one bit-flip error occurs in the shaded region (which denotes the part where we transmit over a noisy channel), then this circuit perfectly corrects it, resulting in the successful transmission of the state $\alpha|0\rangle + \beta|1\rangle$.

It is useful to represent syndrome measurements in terms of stabilisers. For example, a computational basis measurement is represented by the Pauli Z operator. The parity of two qubits is represented by the observable $Z \otimes Z$, since the $Z \otimes Z$ measurement will have outcome $+1$ in the case of even parity and -1 in the case of odd parity (when applied to two of the three qubits). To detect errors in a repetition encoding, we consider the parity of all pairs of qubits in the code; in the case of the three-qubit repetition code, we use the operators

$$Z \otimes Z \otimes \mathbf{1} Z \otimes \mathbf{1} \otimes Z \mathbf{1} \otimes Z \otimes Z.$$

However, since $Z^2 = \mathbf{1}$, it actually suffices to use only two of these, say $Z \otimes Z \otimes \mathbf{1}$ and $\mathbf{1} \otimes Z \otimes Z$, because we can recover the last one as their product. But these two operators are exactly generators of the stabiliser group

$$\mathcal{S} = \{\mathbf{111}, ZZ\mathbf{1}, Z\mathbf{1}Z, \mathbf{1}ZZ\}$$

(where we again drop the tensor product symbol). So, in summary, measuring the two generators of this stabiliser group gives us the error syndrome.

We can compile all the error syndromes (in the case of a single bit-flip) into a table:

Error	$ZZ\mathbf{1}$	$Z\mathbf{1}Z$	$\mathbf{1}ZZ$
$\mathbf{111}$	+	+	+
$X\mathbf{11}$	-	-	+
$\mathbf{1}X\mathbf{1}$	-	+	-
$\mathbf{11}X$	+	-	-

Here the rows are labelled by bit-flip errors, and the columns by the parity-check observables; we write \pm to mean ± 1 . Note how the $+$ and $-$ results correspond to the binary labels 0 and 1, and also how the $Z\mathbf{1}Z$ column is simply given by the product of the other two measurement columns.

13.8 Correcting phase-flips

We have seen how the classical [3, 1, 3] code can be adapted to detect and correct for a single quantum bit-flip, but in Section 13.4 we said that there are three possible errors that we need to worry about: bit-flips, phase-flips, and bit-and-phase flips.

Having dealt with the first, we now deal with the second; finding a way to combine these two solutions to deal with the third is the subject of Section 13.10.

It turns out that we really don't need to do much work in order to solve the problem of single phase-flip errors if we make use of the fact that $HZH = X$, i.e. phase-flips become bit-flips when sandwiched between Hadamards!

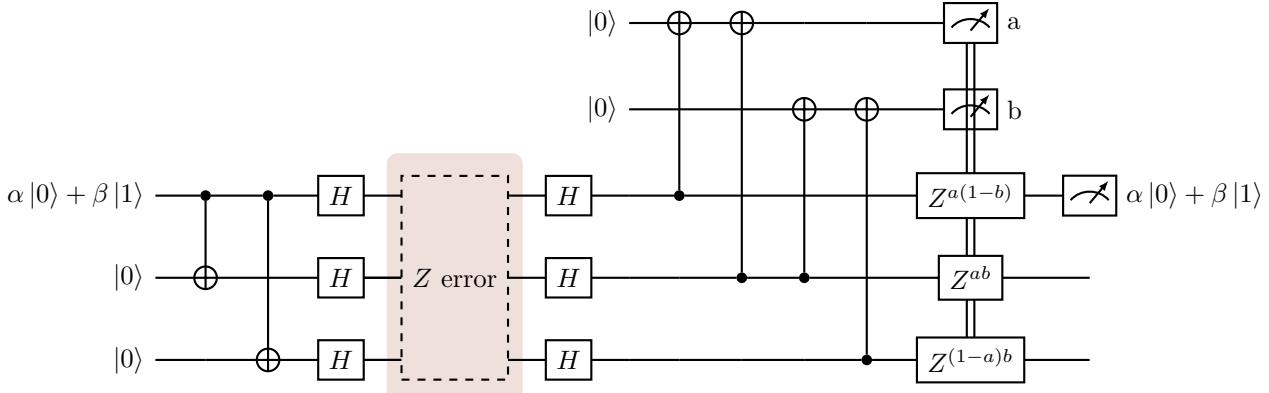


Figure 13.8: Using the quantised [3, 1, 3] code to deal with phase-flips by sandwiching the transmission area between Hadamards.

The encoded state that enters the transmission area affected by decoherence now reads $\alpha|++\rangle + \beta|--\rangle$, where $|\pm\rangle = (|0\rangle \pm |1\rangle)/\sqrt{2}$. These are eigenstates of Z , i.e. $Z|\pm\rangle = |\mp\rangle$, and so errors get transformed into orthogonal, and thus detectable, states.

But just as how the circuit in Section 13.7 only protected against bit-flips, this circuit only protects against phase-flips — now we need to find a way to combine them.

13.9 Composing correctable channels

We have already seen that we can compose quantum channels both in sequence and in parallel, using matrix multiplication or the tensor product (respectively). When we compose two correctable channels, do we still get a correctable channel? Well, if $\{V_i\}$ and $\{W_m\}$ are two sets of channels then

$$(V_j \otimes W_n)^\dagger (V_i \otimes W_m) = (V_j^\dagger V_i) \otimes (W_n^\dagger W_m)$$

and so if the V_i and the W_m are all isometries, then so too is their parallel composition, since the above then equals $\delta_{ij}\delta_{mn} \mathbf{1} \otimes \mathbf{1}$. Similarly, if $V_i: \mathcal{B}(\mathcal{H}) \rightarrow \mathcal{B}(\mathcal{H}')$ and $W_m: \mathcal{B}(\mathcal{H}') \rightarrow \mathcal{B}(\mathcal{H}'')$, then the composition $W_m V_i$ is meaningful, and

$$(W_n V_j)^\dagger (W_m V_i) = V_j^\dagger W_n^\dagger W_m V_i$$

and so if the V_i and W_m are all isometries, then so too is their sequential composition, since the above then equals $\delta_{ij}\delta_{mn} \mathbf{1}$.

Let's apply this to our continuing example of single-qubit error correction. Recall the isometries

$$\begin{aligned} V_{00} &= |000\rangle\langle 0| + |111\rangle\langle 1| \\ V_{01} &= |001\rangle\langle 0| + |110\rangle\langle 1| \\ V_{10} &= |010\rangle\langle 0| + |101\rangle\langle 1| \\ V_{11} &= |100\rangle\langle 0| + |011\rangle\langle 1|. \end{aligned}$$

from Section 13.1. If we write \bar{x} to mean the complement $\text{NOT}(x)$ of a binary string x , then these can all be expressed as

$$V_x = |0x\rangle\langle 0| + |1\bar{x}\rangle\langle 1|.$$

We can define a related set of isometries by

$$W_x = (H \otimes H \otimes H)V_x$$

which correct for any single Z error, using the fact that $HXH = Z$. Then the composites

$$(V_{y_1} \otimes V_{y_2} \otimes V_{y_3})W_x = (V_{y_1} \otimes V_{y_2} \otimes V_{y_3})(H \otimes H \otimes H)V_x$$

define a set of isometries that map a single qubit to nine qubits as a correctable channel.

If we look at the $y_1 = y_2 = y_3 = x = 00$ case, then we can use this to define an encoding procedure, as in Figure 13.9.

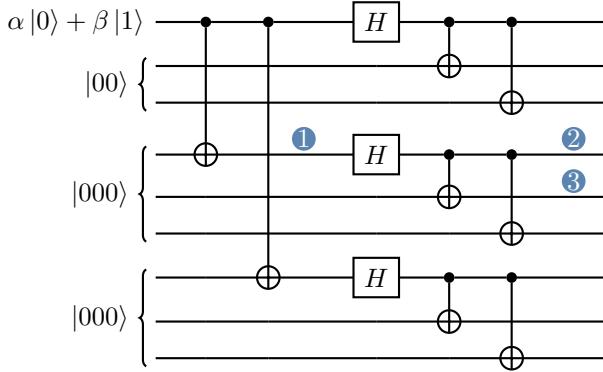


Figure 13.9: The encoding circuit for the Shor $[[9, 1, 3]]$ code, implementing $(V_{00} \otimes V_{00} \otimes V_{00})(H \otimes H \otimes H)V_{00}$. The three locations marked with numbers are not part of the circuit, but we will use them to explain how this circuit corrects for arbitrary single-qubit errors.

What errors can this code cope with? Trivially, an X on any of the nine qubits corresponds to a different isometry, by construction. For example, applying an X at location 3 in Figure 13.9 is picked out by $y_2 = 10$. In other words, each block of three qubits behaves just as it did before.

We also know that, by construction, the code would correct for a single X error at location 1 in the circuit. If we propagate this error through the circuit, this is the same as a Z error in location 2, which corresponds to the isometry with $x = 10$, and shows that Z errors on the first, fourth, and seventh (i.e. the first in each block of three) qubits can be corrected. What about other Z errors? Well, let's go back to the subspace created by one of the V_x , say V_{00} which is spanned by $|000\rangle$ and $|111\rangle$. Then the effect of a single Z error on that space is the same no matter where the Z is applied: a Z error at location 3 has exactly the same effect as a Z error at location 2, since it corresponds to the same isometry, and so the error can still be detected and corrected *without ever needing to know whether the error was at location 2 or location 3*. This lack of knowledge means that the code is said to be **degenerate**.

This circuit gives a nine-qubit encoding that can correct for *any single-qubit* error. The resulting code is called the **Shor $[[9, 1, 3]]$ code**, where the 9 tells us the size of the encoding, the 1 tells us the input size, and the 3 tells us the distance (defined below).

We will describe how the Shor [[9, 1, 3]] code provides single-qubit error correction in more detail in Section 13.10.

Note that, for the Shor [[9, 1, 3]] code, operators such as $Z_1 Z_2 := Z \otimes Z \otimes \mathbf{1}^{\otimes 7}$ are undetectable, but they do not change the logical state:

$$Z_1 Z_2 (V_{y_1} \otimes V_{y_2} \otimes V_{y_3}) W_x = (V_{y_1} \otimes V_{y_2} \otimes V_{y_3}) W_x.$$

However, operators such as $X_1 X_2 X_3$ are undetectable *and do change* the logical state:

$$(X \otimes X \otimes X) V_{y_i} = V_{y_i} X.$$

In fact, this is the smallest possible operator (said to be **of weight 3**, since it is built as a tensor product of three non-trivial gates) that can cause this problem of undetectable but fatal errors, which is exactly the same as saying that the Shor code has distance $d = 3$.

13.10 Correcting any single error: Shor [[9,1,3]]

In Section 13.9 we derived the encoding circuit for the Shor [[9, 1, 3]] code, so now let's go from the top and put all the pieces together to understand how this gives an error correction procedure for all possible single-qubit errors.

To start, we encode our qubit with the phase-flip code

$$\begin{aligned} |0\rangle &\mapsto |+\rangle|+\rangle|+\rangle \\ |1\rangle &\mapsto |-\rangle|-\rangle|-\rangle \end{aligned}$$

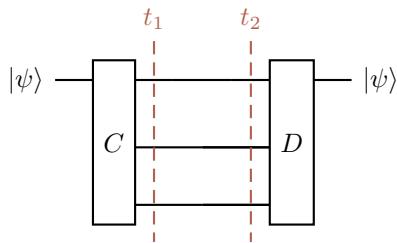
and then we encode each of the resulting three qubits with the bit-flip code

$$\begin{aligned} |0\rangle &\mapsto |0\rangle|0\rangle|0\rangle \\ |1\rangle &\mapsto |1\rangle|1\rangle|1\rangle \end{aligned}$$

resulting in a net effect of

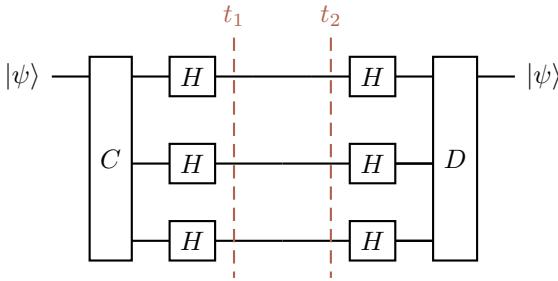
$$\begin{aligned} |0\rangle &\mapsto (|000\rangle + |111\rangle)(|000\rangle + |111\rangle)(|000\rangle + |111\rangle)/\sqrt{8} \\ |1\rangle &\mapsto (|000\rangle - |111\rangle)(|000\rangle - |111\rangle)(|000\rangle - |111\rangle)/\sqrt{8}. \end{aligned} \quad (\ddagger)$$

In order to understand how this code works, it is helpful to look at the complete circuit diagram, so let's build it up in a compositional way. Rather than drawing the entire circuits from Sections 13.7 and 13.8 again, let's simply draw them as consisting of an encoding gate C and a decoding gate D , separated by a zone $[t_1, t_2]$ that corresponds to transmission over the noisy channel. Then the bit-flip correction circuit looks like



and the phase-flip correction looks the same, but with Hadamard gates sandwiching the transmission zone, so

We use double square brackets to emphasise that this is a quantum, not classical, code.



Then we can nest the bit-flip correction circuit into the phase-flip correction circuit by inserting a copy on each of the three wires in the transmission zone, giving us the circuit that implements the encoding of Equation (‡), as shown in Figure 13.10.

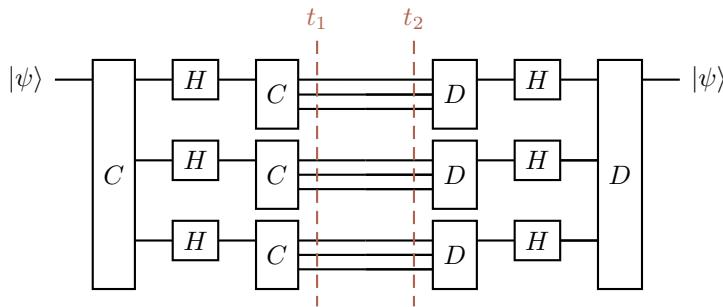


Figure 13.10: Nesting the two correction circuits: one copy of the bit-flip correction circuit on each wire of the phase-flip correction circuit.

Operads.

We have already seen the idea of sequential composition compared to parallel composition, when we talk about the difference between matrix multiplication BA (“do A then B ”) and the tensor product $A \otimes B$ (“do A to one part and B to the other”). Neither type of composition can deal with choices: if A and B are of the right size, then BA (or $A \otimes B$) is either defined or it isn’t. However, there are some subtleties to be aware of.

For example, associativity of a composition operation tells us that we can forget about brackets if we just have the same type of composition over and over, since $(CB)A = C(BA)$. It seems like every operation we ever meet is associative (indeed, it’s even baked into the definition of what it means to be a group), but it turns out that non-associative operations are just as interesting as non-commutative ones. But the story doesn’t stop there! What if we want to study “the next thing up” from associativity, whatever this might be? Or what if we want to study operations that have more than two inputs, and maybe even more than one output? Trying to answer questions like this leads us to **operads** and their algebras.

Looking again at Figure 13.10, we see that we could have composed the bit-flip correction circuits in a different way, placing them one after the other, but we instead wanted to nest them. All that matters in order for us to be able to do either one of these compositions (whether or not we can find any use for it!) is that the number of input and output wires match up. Working with (algebras over) operads has a similar flavour, and you will find yourself drawing lots of little diagrams and then either putting them side-by-side or nesting copies of one inside bits of the other in various different ways. You might find some intriguing pictures if you search for the **little cubes operad**,

or the **Swiss cheese operad**. One particularly nice introduction is Tai-Danae Bradley's "[What is an Operad?](#)".

Now, if an X error occurs on one of the nine qubits in the circuit in Figure 13.10 during the time interval $[t_1, t_2]$, it will be corrected by the corresponding inner three-qubit repetition code that corrects for bit-flips. In fact, this scheme can tolerate up to *three* bit-flip errors *provided that they occur in different blocks*. For example, writing X_i to mean an X error on the i -th qubit, if X_1 , X_5 , and X_7 all occur then they will all be corrected, but if X_1 and X_2 both occur then the resulting error will *not* be corrected.

Next, if a Z error occurs on one of the qubits, say the first one, we know that it will *not* be corrected by the inner encoding-decoding circuit (the one taking place on the top three qubits), but it will be passed along and then corrected "one level up", by the outer encoding-decoding circuit (the one taking place on the first, fourth, and seventh qubits).

Finally, what about if a Y error occurs? Well, since $Y = ZX$, the inner circuit will correct the X part of the error, and the outer circuit will correct the Z part

So quantum error correction is indeed possible: we can remove the unwanted effects of decoherence during transmission through a channel. However, this process of encode-transmit-decode doesn't really cover the practical scenario of computation, since in reality we are constantly trying to process our data, and noise could enter at any moment. One thus has to *compute on the encoded states the whole time*, whilst also somehow ensuring that even faults occurring during an error correction cycle don't adversely affect the computation. This is known as **fault tolerance**, and studying this, using the stabiliser formalisation of Chapter 7, is the goal of Chapter 14.

13.11 Error-correcting assumptions

Throughout this section we have been making certain key assumptions about how errors can occur. For example, we have always assumed that all errors are independent, and that only single-qubit errors can occur. Although this describes many simple scenarios, it does not do a very good job of modelling what happens in practice. It might be the case that many-qubit errors can occur, and that once a specific error has occurred it makes other errors more or less likely. Before we can describe fully fault-tolerant computation, we need to be able to deal with these more complicated scenarios, and this forms the topic of a large chunk of Chapter 14.

However, there are other assumptions that we are making that we will *not* discuss in this text, because they fall out of the scope of "introductory" and instead become the topic of specialised research in error correction and fault tolerance. One such assumption is that we never have any errors affecting our ancilla qubits, so that we can trust our error-syndrome measurements; similarly we assume that when we actually come to apply the error correction, we can do so in an error-free environment. Another assumption is something more "implementation-focused", namely that we are correctly operating the measurement devices, and that they are well-calibrated, otherwise we run into the problem of **measurement errors**. Often combined with this is the problem of **state preparation** — how do we know that we really are preparing the state that we call $|0\rangle$? Such worries, and many more besides, are important if we wish to develop a truly robust theory of error correction.

13.12 Remarks and exercises

13.12.1 Decoherence-free subspaces

Which of the following sets of isometries are correctable?

We will explain why this condition of "occurring in different blocks" is necessary in Section 14.7.

As per usual, any resulting global phase doesn't matter.

The combination of state preparation and measurement errors is sometimes called **SPAM**. This can be dealt with in various ways, appealing to the fact that their effect does not get worse as circuit depth increases since these problems are located at the very beginning and very end of the circuit.

1. $\{V_0, V_1\}$, where

$$\begin{aligned} V_0 &= |00\rangle\langle 0| + |11\rangle\langle 1| \\ V_1 &= \frac{1}{\sqrt{2}} \left[(|01\rangle + |10\rangle)\langle 0| + (|01\rangle - |10\rangle)\langle 1| \right]. \end{aligned}$$

2. $\{V_0, V_1\}$, where

$$\begin{aligned} V_0 &= |00\rangle\langle 0| + |11\rangle\langle 1| \\ V_1 &= \frac{1}{\sqrt{2}} \left[(|01\rangle + |10\rangle)\langle 0| + (|00\rangle - |11\rangle)\langle 1| \right]. \end{aligned}$$

3. $\{U^{\otimes 4}V_0 \mid U \text{ unitary}\}$, where

$$\begin{aligned} V_0 &= \frac{1}{2} \left[(|01\rangle - |10\rangle)(|01\rangle - |10\rangle) \right] \langle 0|. \\ &\quad + \frac{1}{\sqrt{12}} \left[2|0011\rangle + 2|1100\rangle - (|01\rangle + |10\rangle)(|01\rangle + |10\rangle) \right] \langle 1|. \end{aligned}$$

13.12.2 Repetition encoding and majority voting failure

Consider encoding a single classical bit as $2k + 1$ bits using a repetition code, and then decoding with majority voting. If during the transmission process between encoding and decoding each bit is flipped with independent probability p , what is the probability of an error on the logical bit after the encoding–decoding process?

13.12.3 Correcting Pauli rotations with three qubits

We protect an unknown single-qubit state $\alpha|0\rangle + \beta|1\rangle$ against bit-flip errors by encoding it with the three-qubit repetition code:

$$|\psi\rangle = \alpha|000\rangle + \beta|111\rangle.$$

An error of the form $(\cos \theta)\mathbf{1} + (i \sin \theta)X$ occurs on the first qubit during transmission. When we perform the error syndrome measurements, what are the possible outcomes, and what are the corresponding output states?

Conclude that the standard error-correcting protocols that we have discussed will also correct for this type of error.

13.12.4 More on Shor [[9,1,3]]

1. Give the **logical codewords** $|0_L\rangle$ and $|1_L\rangle$ for the Shor [[9, 1, 3]] code.
2. What is the smallest number of single-qubit operations needed to convert $|0_L\rangle$ into $|1_L\rangle$?
3. Can you identify the stabilisers and the **logical operators** X_L and Z_L for this code? Note that these may not be unique.
4. Write a table of the syndromes for all single-qubit X or Z errors on this code, where the columns are labelled by the single-qubit error, and the row by the corresponding stabiliser.
5. How can we detect and correct a Y error occurring on the first qubit?
6. If an error of the form $\sqrt{1-p}\mathbf{1} + i\sqrt{p}Y$ occurs on the first qubit, what are the different possible outcomes of measurement?

That is, the states corresponding to the encoding of $|0\rangle$ and $|1\rangle$.

That is, the operators X_L and Z_L that behave on $|0\rangle_L$ and $|1\rangle_L$ exactly how X and Z behave on $|0\rangle$ and $|1\rangle$. Hint: start from the encoding circuit with the eight ancillas all prepared in state $|0\rangle$; what are their stabilisers? Recall that the encoding operation is a Clifford circuit.

7. Assume that there is some environment, initially in state $|e\rangle$. Decoherence occurs on the qubit, transforming it via

$$\begin{aligned}|0\rangle|e\rangle &\mapsto |0\rangle|e_{00}\rangle \\|1\rangle|e\rangle &\mapsto |0\rangle|e_{11}\rangle.\end{aligned}$$

Show that, if we use the Shor $[[9, 1, 3]]$ code and this decoherence only affects the first qubit in transmission, then we can correct for the resulting error.

13.12.5 Distillation for Bell pairs

Alice wants to send m qubits of information to Bob. She can send quantum states, but only through a transmission channel that induces errors, though she can send classical information perfectly. Bob cannot send messages (neither quantum nor classical) to Alice, but both of them can perfectly implement quantum logic gates.

To send her m qubits in spite of the noise, Alice might encode them in an n -qubit error correcting code.

The process by which a set of N noisy Bell pairs is converted into a small number M of perfect Bell pairs is known as **distillation**. This occurs at a rate $D_1 = M/N$, which is often considered in the limit of large N . The subscript 1 denotes that this is **one-way** distillation, where only Alice can send messages.

- Assuming knowledge of the optimal code (i.e. one that is guaranteed to succeed and is as small as possible), Alice could transmit encoded halves of Bell pairs, which Bob could then decode. What is a bound on the rate at which Alice and Bob can distill Bell pairs through this channel?
- Alternatively, Alice could send Bob unencoded halves of Bell pairs, which they then distill to create a smaller number of perfect Bell pairs which Alice can then use to teleport the desired information. Assuming knowledge of the optimal distillation procedure (i.e. one that maximises D_1), how does this protocol bound the distillation rate?

13.12.6 Composing quantum codes

Consider two quantum codes: C_1 is an $[[n_1, 1, d_1]]$ code, and C_2 is an $[[n_2, 1, d_2]]$ code. We decide to encode a qubit $|\psi\rangle$ by first encoding it into n_1 qubits using C_1 , and then encoding each of those resulting qubits into n_2 qubits using C_2 . The overall effect is an encoding into the composite code C_2C_1 .

- How many physical qubits are involved in the encoding of a single logical qubit of the new code?
- What is the distance of the new code?

14 Quantum error correction

About more classical error correction and the **Hamming codes**, how they generalise to quantum codes called **CSS codes**, including the **Steane [[7,1,3]]-code**. Also about computing in the presence of errors: **logical states**, **logical operators**, and **error families** — all via the formalism of stabilisers — and the **transversal gates** that we can implement to act on them.

We have seen a way of dealing with the computational errors introduced by the physical problem of decoherence, namely the Shor $[[9, 1, 3]]$ code, but this is just the start of the story. There is a vast body of work on *classical* error correction, so it's sensible to ask if we can adapt this to help us in the world of quantum computation. As we shall see, we can actually use quite a lot of the theory of classical error-correction codes, and in doing so we will start to really make use of the stabiliser formalism introduced all the way back in Chapter 7. But note that this *still* isn't the end of the story: our goal is so-called **fault-tolerant computation**, which we come to in Chapter 15.

14.1 The Hamming code

The challenge in designing efficient error-correcting codes resides in the trade-off between *rate* and *distance* (introduced in Section 13.6). Ideally, both quantities should be high: a high rate signifies low overhead in the encoding process (i.e. requiring only a few redundant bits), and a high distance means that many errors can be corrected. So can we optimise both of these quantities simultaneously? Unfortunately, various established bounds tell us that there is always a trade off, so high-rate codes must have low distance, and high-distance codes must have a low rate. Still, there is a lot of ingenuity that goes into designing good error-correction codes, and some are still better than others!

Before looking at quantum codes in more depth, we again start with *classical* codes. For example, in Section 13.6 we saw the three-bit repetition code, which has a rate of $R = 1/3$ and distance 3. However, the **Hamming $[7, 4, 3]$ code** has the same distance, but a better rate of $R = 4/7 > 1/3$. Figure 14.1 show diagrammatic representations of this Hamming code, which we will now study further.

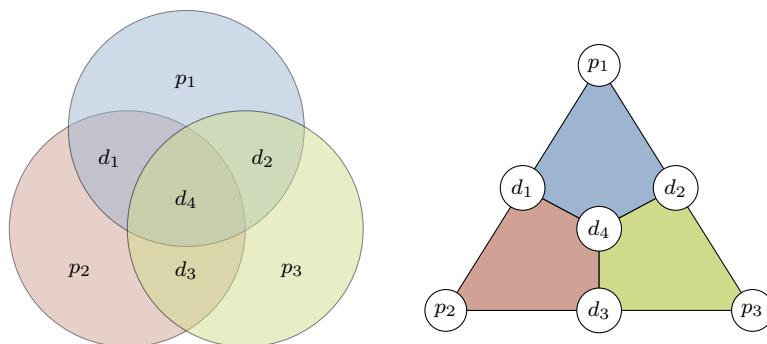
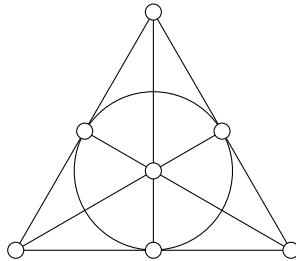


Figure 14.1: *Left:* The Venn diagram for the Hamming $[7, 4, 3]$ code. *Right:* The **plaquette** (or **finite projective plane**) diagram for the same code. In both, d_i are the **data bits** and the p_i are the **parity bits**. The coloured circles (resp. coloured quadrilaterals) are called **plaquettes**.

In the late 1940s, Richard Hamming, working at Bell Labs, was exasperated by the fact that the machines running his punch cards (in these heroic times of computer science, punch cards were the state of the art data storage) were good enough to notice when there was an error (and halting) but not good enough to know how to fix it.

The Fano plane.

We say that the plaquette diagram in Figure 14.1 could also be called a *finite projective plane* diagram because of how it resembles the [Fano plane](#), which is the [projective space of dimension 2](#) over the [field with 2 elements](#).



In fact, there is more than a mere visual similarity between these two diagrams: we will soon introduce the formal definition of a [linear code](#), and there is a special family of these known as [projective codes](#), which are those such that the columns of the generator matrix (another character who we shall soon meet) are all distinct and non-zero.

Projective codes are particularly interesting because they allow us to apply geometric methods to study the properties of the code. For example, the columns of the parity check matrix of a projective code correspond to points in some projective space. Furthermore, since the geometry in question concerns *finite dimensional* spaces over *finite* fields, we end up coming across a lot of familiar (and useful) combinatorics. This is partially due to the fact that finite geometry can be understood as an example of an [incidence structure](#).

Both diagrams in Figure 14.1 describe the same situation, but although the right-hand one is useful for understanding the geometry hidden in the construction and allowing us to generalise to create new codes, and is thus the one that we will tend to use, the Venn diagram on the left-hand side is maybe more suggestive of what's going on.

The idea is that we have a four-bit string $d_1d_2d_3d_4$ consisting of the four **data bits**, and we encode into a seven-bit string $d_1d_2d_3d_4p_1p_2p_3$ by appending three **parity bits** p_1 , p_2 , and p_3 , which are defined by

$$p_1 = d_1 + d_2 + d_4 \bmod 2$$

$$p_2 = d_1 + d_3 + d_4 \bmod 2$$

$$p_3 = d_2 + d_3 + d_4 \bmod 2.$$

You can hopefully see how the triangular diagram in Figure 14.1 tells us which data bits are used in defining each parity bit: we take the sum of all data bits in the same **plaquette** (one of the three coloured quadrilaterals) as the parity bit.

We can also express this encoding in matrix notation, defining the **data vector** \mathbf{d} by

$$\mathbf{d} = \begin{bmatrix} d_1 \\ d_2 \\ d_3 \\ d_4 \end{bmatrix}$$

and the **generator matrix** G by

Sometimes you will see the Hamming $[7, 4, 3]$ code referred to simply as the **seven-bit Hamming code**, the $[7, 4, 3]$ code, or even just **the Hamming code**.

Many sources define G to be the transpose of what we use here, but this is just a question of convention. Because of this, we'll be dealing with the *column* (not row) spaces of matrices, also known as the **range**.

$$G = \left[\begin{array}{cccc} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \\ \hline 1 & 1 & 0 & 1 \\ 1 & 0 & 1 & 1 \\ 0 & 1 & 1 & 1 \end{array} \right]$$

The vector space spanned by the columns of the generator matrix G is known as the **codespace** of the code, and any vector in this space is known as a **codeword**.

The encoding process is then given by the matrix G acting on the vector \mathbf{d} . Indeed, since the top (4×4) part of G is the identity, the first four rows of the output vector $G\mathbf{d}$ will simply be a copy of \mathbf{d} ; the bottom (3×4) part of G is chosen precisely so that the last three rows of $G\mathbf{d}$ will be exactly p_1 , p_2 , and p_3 . In other words,

$$G\mathbf{d} = \begin{bmatrix} d_1 \\ d_2 \\ d_3 \\ d_4 \\ p_1 \\ p_2 \\ p_3 \end{bmatrix} = \begin{bmatrix} \mathbf{d} \\ p_1 \\ p_2 \\ p_3 \end{bmatrix}.$$

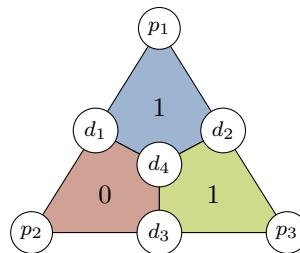
By construction, the sum of the four bits in any single plaquette of the code sum to zero. For example, in the bottom-left (red) plaquette,

$$\begin{aligned} p_2 + d_1 + d_3 + d_4 &= d_1 + d_3 + d_4 + d_1 + d_3 + d_4 \\ &= 2(d_1 + d_3 + d_4) \\ &= 0 \end{aligned}$$

Since we are working with classical bits, all addition is taken mod 2, so sometimes we will neglect to say this explicitly.

and the same argument holds for the other two plaquettes. This incredibly simple fact is where the power of the Hamming code lies, since it tells the receiver of the encoded string a lot of information about potential errors.

Let's consider a concrete example. Say that Alice encodes her data string $d_1 d_2 d_3 d_4$ and sends the result $G\mathbf{d}$ to Bob, who takes this vector and looks at the sum of the bits in each plaquette, and obtains the following:



We don't write the values of the bits, only the sums of the bits in each plaquette. This is because we don't need to know the value of the bits in order to know where the error is, only the three sums!

If we make the assumption that at most one error occurs then this result tells us exactly where the bit-flip happened: it is *not* in the bottom-left (red) plaquette, but it *is* in both the top (blue) and bottom-right (yellow) plaquettes. Looking at the diagram we see that it must be d_2 that was flipped, and so we can correct for this error by simply flipping it back before unencoding (where the unencoding process is given by simply forgetting the last three bits of the received string).

We can describe the error location process in terms of matrices as well, using the **parity-check matrix** H , given by

$$H = \left[\begin{array}{cccc|ccc} 1 & 1 & 0 & 1 & 1 & 0 & 0 \\ 1 & 0 & 1 & 1 & 0 & 1 & 0 \\ 0 & 1 & 1 & 1 & 0 & 0 & 1 \end{array} \right].$$

This assumption is crucial here, and we investigate what happens if we drop it in Section 14.7.

Here is yet another H , to go with the Hadamard and the Hamiltonian...

Note that the rows of H are exactly the coefficients of the parity-check equations for each plaquette, where we order them top-left-right (blue-red-yellow). For example, to get the sum corresponding to the bottom-left (red) plaquette, we need to sum the first, third, fourth, and sixth bits of the encoded string $d_1d_2d_3d_4p_1p_2p_3$, and these are exactly the non-zero entries of the second row of H . The columns of the parity-check matrix H are known as the **error syndromes**, for reasons we will now explain

The parity-check matrix H is defined exactly so that

$$H\mathbf{c} = 0 \iff \mathbf{c} \text{ is a codeword.}$$

Now we can see a bit more into how things work, since linearity of matrix multiplication tells us that, if a receiver receives $\mathbf{c} + \mathbf{e}$ where \mathbf{e} is the error,

$$\begin{aligned} H(\mathbf{c} + \mathbf{e}) &= H\mathbf{c} + H\mathbf{e} \\ &= H\mathbf{e}. \end{aligned}$$

Decoding the message then consists of finding the most probable error \mathbf{e} that yields the output $H\mathbf{e}$. If \mathbf{e} is a single bit-flip error, then $H\mathbf{e}$ is exactly a column of H , which justifies us describing the columns as error syndromes. We can construct a table describing all of the possible error syndromes, and which bit they indicate for us to correct:

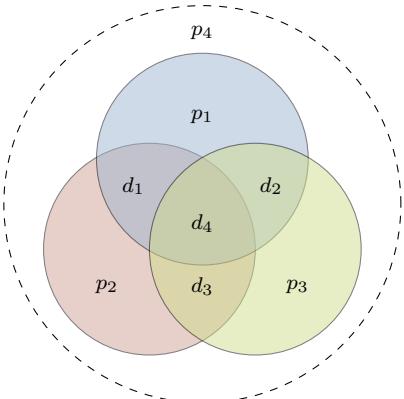
Syndrome	000	110	101	011	111	100	010	001
Correction	-	d_1	d_2	d_3	d_4	p_1	p_2	p_3

The above construction of the Hamming $[7, 4, 3]$ code can be generalised to result in a Hamming $[2^r - 1, 2^r - r - 1, 3]$ code for any $r \geq 2$, where each column of the parity-check matrix is a different binary string, excluding the string of all 0 bits. It's noteworthy that only a logarithmic number of parity checks are necessary to correct all single-bit errors. However, there are some downsides to Hamming codes. Although the rate $R = (2^r - r - 1)/(2^r - 1)$ approaches 1 as $r \rightarrow \infty$, Hamming codes are impractical in highly noisy environments because they have a fixed distance of 3.

Recall that codewords are exactly those vectors of the form $G\mathbf{d}$ for some data vector \mathbf{d}

Double-bit errors in the Hamming code.

Although the Hamming $[7, 4, 3]$ code can only deal with single-bit errors, it can be extended to an $[8, 4, 4]$ code, at least *detecting* double-bit errors, by adding a single extra parity bit p_4 , given by taking the sum of all the other seven bits:



How do we know that the distance is always 3? Well, there are *triples* of columns in the parity-check matrix that, when added together, give all zeros. This means that there are sets of 3 errors such that, if they all occur together, the syndrome will be zero, and so the distance is no more than 3. Meanwhile, all the columns are distinct, so no pair of columns add together trivially, which means that the distance must be greater than 2.

How does this work? Well, let's assume that up to two bit-flip errors could occur. The decoding process then starts by looking at this new parity bit p_4 in the received string and seeing if it is indeed equal to the sum of all the other

bits, saying that it is “correct” if so, and “incorrect” if not. If p_4 is *incorrect*, then there has been a *single-bit* error, and we can just look at the rest of the string $d_1d_2d_3d_4p_1p_2p_3$ and apply the previous Hamming [7, 4, 3] code decoding process, with the caveat that if this tells us that no errors have occurred, then it must be the case that the single-bit error actually flipped the parity bit p_4 itself. If p_4 is *correct*, then there has either been no error or a double bit-flip error; to see which is the case we can measure the Hamming [7, 4, 3] code error syndrome of $d_1d_2d_3d_4p_1p_2p_3$, and this will tell us the XOR of the two bit-flip locations; if this is 0 then either no error has occurred or two errors affected the same bit, cancelling each other out.

Before moving on, it will be useful to introduce another common way of diagrammatically representing parity-check matrices called a **Tanner graph**. This is a bipartite graph consisting of two types of nodes: the **codeword** (or **data**) nodes (one for each bit of the codeword, drawn as circles), and the **syndrome** nodes (one for each bit of the syndrome, drawn as squares). The edges in the Tanner graph are such that the parity-check matrix H is exactly the **adjacency matrix** of the graph, i.e. the matrix that has a 1 in the (i, j) -th position if the i -th syndrome node is connected to the j -th codeword node, and a 0 otherwise.

A graph (which is a collection of **nodes** and **edges** between some of them) is **bipartite** if the nodes can be split into two sets such that all the edges go from one element of the first set to one element of the second.

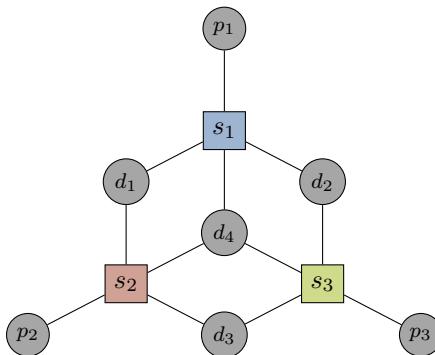


Figure 14.2: The Tanner graph for the Hamming [7, 4, 3] code. Comparing this to the plaquette diagram in Figure 14.1, we see that we simply replace plaquettes by syndrome nodes (hence our choice of colours). It is not immediately evident that this graph is bipartite, but try drawing it with all the syndrome nodes in one row and all the data nodes in another row to see that it is.

One particularly useful aspect of Tanner graphs is how simple it is to convert to and from the corresponding parity-check quantum circuits. There is a syndrome node for each ancilla qubit, and a data node for each data qubit; there are paths between syndrome and data nodes whenever there is a controlled-NOT between the corresponding qubits. We show a simple example in Figure 14.3.

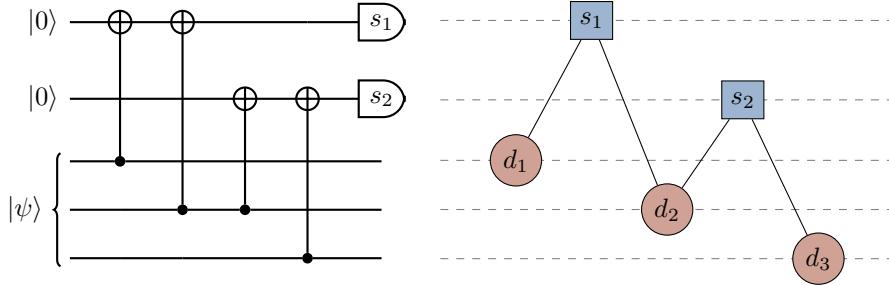


Figure 14.3: *Left:* The quantum circuit for a parity-check operation. *Right:* The corresponding Tanner graph.

14.2 Linear codes

Hamming codes are special cases of a larger family of codes called **linear codes**: one in which the codewords form a vector space. These are constructed by judicious choices of the generators matrices or parity-check matrices (since one determines the other), and can offer different trade-offs between the code rates and distances. We have already seen the example of the Hamming $[7, 4, 3]$ code, but let's state the general framework a bit more abstractly.

An $[n, k, d]$ **linear code** C is described by two matrices:

- The **generator** matrix G , which is an $(n \times k)$ matrix

$$G = \begin{bmatrix} \mathbf{1}_{k \times k} \\ P \end{bmatrix}$$

for some $((n - k) \times k)$ matrix P . The columns of G are **codewords**, and and form a basis for the **codespace** $\text{range}(G)$.

- The **parity-check** matrix H , which is an $((n - k) \times n)$ matrix

$$H = [Q \quad \mathbf{1}_{(n-k) \times (n-k)}]$$

for some $((n - k) \times k)$ matrix Q . The columns of H are **error syndromes**.

The matrices G and H have to satisfy one of the two equivalent conditions

$$\text{range}(G) = \ker(H) \quad \text{or} \quad \text{range}(H^T) = \ker(G^T).$$

We can ensure this by taking $Q = -P$ (see Exercise 14.11.4).

One last piece of the puzzle that we need to understand is the notion of **dual codes**. We will write $\text{range}(G)$ to mean the vector space spanned by the columns of the matrix G . Given a code $C = (G, H)$ expressed in terms of its generator matrix and parity-check matrix, we know that the columns of G span the kernel of H , i.e. that $\text{range}(G) = \ker(H)$. Why is this? Well, because this is equivalent to saying that a vector is in the span of the columns of G exactly when it is of the form Gd for some data vector d , i.e. exactly when it is a codeword, and we have already shown this above. In particular then,

$$H \cdot G = 0$$

(which merely says that $\text{range}(G) \subseteq \ker(H)$).

But, taking the transpose of the above equality, we see that

$$G^T \cdot H^T = 0$$

Usually, for linear codes, people talk of the code C as being equal to the codespace, i.e. the span of the columns of G . For now, however, it is notationally simpler to denote a code by these two key matrices.

and so we can define a code $C^\perp = (H^T, G^T)$, whose codewords are exactly the columns of H^T , i.e. the rows of H . This is known as the **dual code** of C . Since G has n rows and k columns, and H has $n - k$ rows and n columns, we see that the dimension of the codespace of C (i.e. the span of the columns of G) and the dimension of the codespace of C^\perp (i.e. the span of the rows of H) must sum to n . In fact, if C is an $[n, k, d]$ code, then C^\perp is an $[n, n - k, d']$ code, where d' is not usually related to d in any obvious way. A nice example is the Hamming $[7, 4, 3]$ code, whose dual is a $[7, 3, 4]$ code known as the **shortened Hadamard code**.

Given a code $C = (G, H)$, its **dual code** is $C^\perp = (H^T, G^T)$.

This tells us that $\text{range}(H^T) \subseteq \ker(G^T)$, but we can show that this is an equality using the fact that $\text{range}(G) = \ker(H)$ is an equality.

It is immediate that $(C^\perp)^\perp = C$, but it is interesting to ask about the relationship between C and C^\perp . Then we say that a code is **weakly self-dual** (or **self-orthogonal**) if $\text{range}(G) \subseteq \text{range}(H^T)$, and **self-dual** if $\text{range}(G) = \text{range}(H^T)$. In other words, a code is weakly-self dual if its codespace is a subspace of the codespace of its dual, and self-dual if these two codespaces are actually equal. We said above that $\dim \text{range}(G) + \dim \text{range}(H^T) = n$, so we see that for a code to be self-dual it must be the case that n is even, but this is only a necessary condition, not a sufficient one!

14.3 Quantum codes from classical

We would like to use the insights gained from our study of classical codes to help us build quantum codes. Let's start with a classical $[n, k, d]$ code (such as the Hamming $[7, 4, 3]$), with parity-check matrix H and generator G . Each row r of H is a binary string $x_r = x_{r,1}x_{r,2}\dots x_{r,n}$, where $x_{i,j}$ is the (i, j) -th element of H . For $1 \leq r \leq n$, we define a stabiliser generator

$$G_r := X_{x_r} := \otimes_{j=1}^n X^{x_{r,j}}.$$

For example, in the case of the Hamming $[7, 4, 3]$ code, we have

$$H = \left[\begin{array}{cccc|ccc} 1 & 1 & 0 & 1 & 1 & 0 & 0 \\ 1 & 0 & 1 & 1 & 0 & 1 & 0 \\ 0 & 1 & 1 & 1 & 0 & 0 & 1 \end{array} \right].$$

so the three rows define three generators

$$\begin{aligned} G_1 &= X_{1101100} = XX\mathbf{1}XX\mathbf{1}\mathbf{1} \\ G_2 &= X_{1011010} = X\mathbf{1}XX\mathbf{1}X\mathbf{1} \\ G_3 &= X_{0111001} = \mathbf{1}XXX\mathbf{1}\mathbf{1}X. \end{aligned}$$

Now consider a state $|\psi\rangle$ that is stabilised by these generators, i.e. such that

$$G_r |\psi\rangle = |\psi\rangle.$$

What happens if a Z error occurs on a particular qubit: what measurement results do we get when we measure the stabilisers? Well, writing Z_j to mean a Z error on the j -th qubit, as usual,

$$\begin{aligned} G_r Z_j |\psi\rangle &= (-1)^{x_{r,j}} G_r |\psi\rangle \\ &= (-1)^{x_{r,j}} |\psi\rangle. \end{aligned}$$

So the measurement outcome directly corresponds to the (r, j) -th entry $x_{r,j}$ of the parity check matrix. Generally, if this Z_j error occurs, then measuring for all rows r will give measurement outcomes that directly correspond to the j -th column of the

parity check matrix. This is just the same lookup table as in the classical case: this codespace is a distance d error correcting code for single Z errors. Using the Hamming $[7, 4, 3]$ code as an example again, we get the following table of error syndromes, where we write \pm to mean ± 1 :

Error	G_1 outcome	G_2 outcome	G_3 outcome
none	+	+	+
Z_1	-	-	+
Z_2	-	+	-
Z_3	+	-	-
Z_4	-	-	-
Z_5	-	+	+
Z_6	+	-	+
Z_7	+	+	-

So if we measure the three stabilisers and get the measurement sequence $(-1, 1, 1)$ then the corresponding bit string $(1, 0, 0)$ in the Hamming $[7, 4, 3]$ code tells us that there was an error on p_1 , i.e. a Z_5 error.

We have used a classical code to help us correct for Z errors in the quantum case. If we take a second classical code, with parameters $[n, k', d']$ and parity-check matrix H' , and use it to define Z -type stabilisers $G'_r = Z_{x_r}$ then we will have a distance d' protection against X errors. However, we cannot simply pick the two classical codes arbitrary: if the scheme is to work, then the X -type and Z -type stabilisers must actually *be* stabilisers, i.e. they must commute.

The challenge in creating quantum error-correction codes often lies in finding good commuting sets of stabilisers.

How can we tell if this happens? Well, if x is a row of H , and z a row of H' , then we need the number of positions $i \in \{1, \dots, n\}$ such that $x_i = z_i = 1$ to be even, as these are the ones that will give operators that individually anticommute ($XZ = -ZX$). In notation, this is the same as asking that

$$x \cdot z \equiv 0 \pmod{2}.$$

This is the same as saying that z , which is a row of H' , must be a codeword of the first code: by definition of the parity-check matrix H , we need that $Hz = 0$. Applying this reasoning to all rows z , we see that we need

$$H \cdot H'^T = 0$$

or, in other words,

$$\text{range}(H'^T) \subseteq \ker(H).$$

But we know that $\text{range}(G) = \ker(H)$, and so this is equivalent to asking for

$$\text{range}(H'^T) \subseteq \text{range}(G).$$

We can figure out some key properties of combining an $[n, k, d]$ code (G, H) for X -stabilisers and an $[n, k', d']$ code (G', H') for Z -stabilisers without too much difficulty. Since our first code encodes k bits, the generator G has k rows, and the parity-check matrix H has $n - k$ rows. Thus there are $n - k$ of the X -type generators, and $n - k'$ of the Z -type generators; in total there are $2n - (k + k')$ generators. Since each generator halves the dimension, the dimension of the Hilbert space defined by the stabilisers is

$$2^{n-2n+k+k'} = 2^{k+k'-n}$$

i.e. it encodes $k + k' - n$ qubits. The combined code has a distance k against Z errors, and k' against X errors; since the two types of errors are correctly independently, the total distance is simply $\min(d, d')$. In summary then, we have created an

$$[[n, k + k' - n, \min(d, d')]]$$

quantum error-correction code, and its decoding is well understood based on the classical decoding methods applied independently for X errors and Z errors. This general construction of quantum error correcting codes is known as the **CSS construction**, for its originators [Robert Calderbank](#), [Peter Shor](#), and [Andrew Steane](#).

Given an $[n, k_1, d_1]$ code $C_1 = (G_1, H_1)$ and an $[n, k_2, d_2]$ code $C_2 = (G_2, H_2)$ such that $\text{range}(H_2^T) \subseteq \text{range}(G_1)$, the **CSS code** $\text{CSS}(C_1, C_2)$ constructed as above is an $[[n, k_1 + k_2 - n, \min(d_1, d_2)]]$ code.

As always, one needs to be careful of conventions. Many sources define a code to be the codespace range G itself instead of the pair (G, H) , and usually also replace C_2 with C_2^\perp in the statement of the CSS construction.

Before moving on, let's look at the remaining details of applying the CSS construction to the Hamming $[7, 4, 3]$ code. Let $C_1 = C_2 = (G, H)$ be the Hamming $[7, 4, 3]$ code with G and H as in Section 14.1. To apply the CSS construction, we need to check that $\text{range}(H_2^T) \subseteq \text{range}(G_1)$. Since $C_1 = C_2$, this is simply asking that the Hamming $[7, 4, 3]$ code be weakly self-dual, i.e. that

$$\text{range}(H^T) \subseteq \text{range}(G)$$

which can be checked to be true by hand. This means that we can use the Hamming $[7, 4, 3]$ code to define both our X -type and Z -type generators: using the notation of Section 7.2, the group of generators is generated by

$$+ \left| \begin{array}{ccccccc} X & X & 1 & X & X & 1 & 1 \\ X & 1 & X & X & 1 & X & 1 \\ 1 & X & X & X & 1 & 1 & X \\ Z & Z & 1 & Z & Z & 1 & 1 \\ Z & 1 & Z & Z & 1 & Z & 1 \\ 1 & Z & Z & Z & 1 & 1 & Z \end{array} \right|$$

The result is a $[[7, 1, 3]]$ code, generally attributed to and thus known as the **Steane code**, or simply the **seven-qubit code**, that encodes one logical qubit across seven physical ones, and that is able to correct for any single-qubit Pauli error. We can visualise the Steane code using its Tanner graph, as in Figure 14.4, but we will return to a proper in-depth study of this code in Section 14.9.

In particular, what we have defined would often be called the CSS construction of C_1 over C_2^\perp (instead of over C_2).

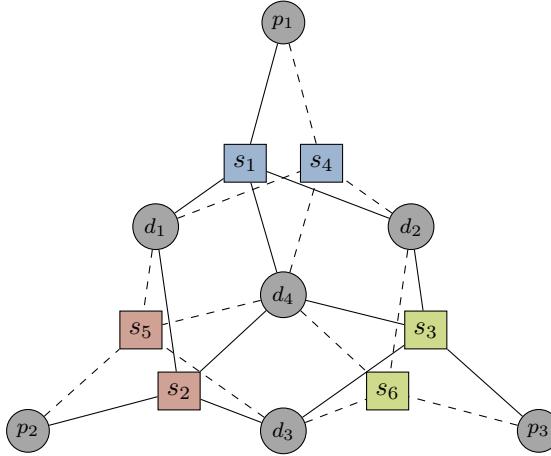


Figure 14.4: The Tanner graph of the Steane code. You can think of this graph as taking two copies of the Hamming [7, 4, 3] code Tanner graph and gluing them together at all of the data nodes. For any CSS code there are two types of “parity checks”: one for detecting X errors (solid lines), and one for Z errors (dashed lines).

Not all quantum codes arise from combining classical ones like this, and even for those that do, working with the generator and parity-check matrices can often be cumbersome. Indeed, a truly *quantum* code will not have a single one of each, since this is not sufficient to deal with the purely quantum phenomena of superposition. For example, as the Tanner graph in Figure 14.4 shows, we have *two* parity check matrices in the case of CSS codes. When working with truly quantum codes, the stabiliser formalism really becomes much more useful — our next aim is to justify this with some examples and explanation.

For example, the **five-qubit code**, which is the smallest code that can correct for all possible single-qubit errors, is demonstrably *not* a CSS code, as can be shown by the non-existence of something called a **transversal controlled-NOT** gate (we discuss this in Section ??).

14.4 Logical operators ...

Here we are going to use the abstract group theory that we developed back in Sections 7.5 and 7.6, but there are other ways of explaining this material. In Section @{logical-operators-differently} we tell the same story from a different point of view, so if you find this section confusing then don’t worry — you can always come back to it after reading the other one! It’s always a good idea to have multiple viewpoints.

We have been slowly building up towards constructing quantum error-correction codes using the stabiliser formalism, but there is one major detail that we have yet to mention. You will perhaps have noticed that we haven’t written out what the stabiliser states actually *are*, nor what the encoding circuits look like. There is a simple reason for this: at this point, we don’t actually know! There’s a little more work to be done — the stabilisers have provided us with a two-dimensional space, but if we have $|0\rangle$ and $|1\rangle$ to encode, how are they mapped within the space? So far, it’s undefined, and there is a lot of freedom to choose, but the structures provided by group theory are quite helpful here in providing some natural choices. Furthermore, better understanding these structures is the first step towards figuring out how to upgrade from simple error correction to fault-tolerant computation. We’re going to turn back all the way to Sections 7.5 and 7.6, where we discovered how to think about normalisers of stabiliser groups inside the Pauli group. Let’s start with a brief recap.

The n -qubit Pauli group \mathcal{P}_n consists of all n -fold tensor products of Pauli matrices $1, X, Y$, and Z , with possible global phase factors ± 1 and $\pm i$. Given an operator $s \in \mathcal{P}_n$, we say that it **stabilises** a (non-zero) n -qubit state $|\psi\rangle$ if $s|\psi\rangle = |\psi\rangle$, i.e. if it admits $|\psi\rangle$ as an eigenstate with eigenvalue $+1$. We showed that the set of all operators that stabilise every state in a given subspace V form a group, called the **stabiliser group**; using a little bit of group theory, we characterised all possible stabiliser groups by showing that they are exactly the abelian subgroups of \mathcal{P}_n that do not contain -1 .

Then we looked at the group structure of the Pauli group, and how any stabiliser group \mathcal{S} sits inside it. It turned out that the **normaliser**

$$N(\mathcal{S}) = \{g \in \mathcal{P}_n \mid gsg^{-1} \in \mathcal{S} \text{ for all } s \in \mathcal{S}\}$$

of \mathcal{S} in \mathcal{P}_n , and the **centraliser**

$$Z(\mathcal{S}) = \{g \in \mathcal{P}_n \mid gsg^{-1} = s \text{ for all } s \in \mathcal{S}\}$$

of \mathcal{S} in \mathcal{P}_n actually agree, because of some elementary properties of the Pauli group. Furthermore, we showed that the normaliser (or centraliser) was itself normal inside the Pauli group, giving us a chain of normal subgroups

$$\mathcal{S} \triangleleft N(\mathcal{S}) \triangleleft \mathcal{P}_n.$$

This lets us arrange the elements of \mathcal{P}_n into cosets by using the two quotient groups

$$N(\mathcal{S})/\mathcal{S} \quad \text{and} \quad \mathcal{P}_n/N(\mathcal{S}).$$

How does this help us with our stabiliser error-correction codes? Let's look first at the former: cosets of \mathcal{S} inside its normaliser $N(\mathcal{S})$.

If $|\psi\rangle \in V_{\mathcal{S}}$ is a state in the stabilised subspace, then any element $g \in \mathcal{S}$ always satisfies

$$g|\psi\rangle = |\psi\rangle$$

whereas any element $g \in N(\mathcal{S}) \setminus \mathcal{S}$ merely satisfies

$$g|\psi\rangle \in V_{\mathcal{S}}$$

and, for any such g , there are always states in $V_{\mathcal{S}}$ that are not mapped to themselves. However, if we look at cosets of \mathcal{S} inside $N(\mathcal{S})$ then we discover an incredibly useful fact: all elements of a given coset act on $|\psi\rangle$ in the same way. To see this, take two representatives for a coset, say $g\mathcal{S} = g'\mathcal{S}$ for $g, g' \in N(\mathcal{S})$. By the definition of cosets, this means that there exist $s, s' \in \mathcal{S}$ such that $gs = g's'$. In particular then,

$$gs|\psi\rangle = g's'|\psi\rangle$$

but since $s, s' \in \mathcal{S}$ and $|\psi\rangle \in V_{\mathcal{S}}$, this says that

$$g|\psi\rangle = g'|\psi\rangle$$

as claimed.

Since the cosets of \mathcal{S} inside $N(\mathcal{S})$ give well defined actions on stabiliser states, preserving the codespace, we can treat them as operators in their own right.

The cosets of \mathcal{S} inside $N(\mathcal{S})$ are called **logical operators**, and any representative of a coset is an **implementation** for that logical operator.

Let's try to understand this in the context of an example: the three-qubit code from Section 13.1. The diagram from Section 7.2 was useful in describing this example, so we repeat it as Figure 14.5 below.

For us here, the stabilised subspace $V_{\mathcal{S}}$ is exactly the codespace, and the stabilisers generating \mathcal{S} are exactly the elements $G_r \in \mathcal{P}_n$ constructed from the rows of H as at the start of Section 14.3.

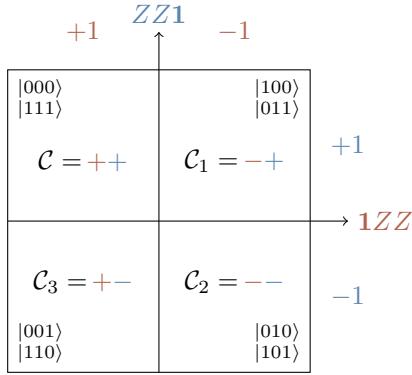


Figure 14.5: The stabiliser group $\mathcal{S} = \langle ZZ1, 1ZZ \rangle$ bisects the Hilbert space of three qubits into four equal parts, and gives the stabilised subspace V_S which is spanned by $|000\rangle$ and $|111\rangle$.

To use the terminology of error-correction codes, we are taking our codespace to be

$$\mathcal{C} = \langle |000\rangle, |111\rangle \rangle$$

which is exactly the stabiliser space V_S of the stabiliser group

$$\mathcal{S} = \langle ZZ1, 1ZZ \rangle$$

and the total eight-dimensional Hilbert space of three qubits is decomposed into four mutually orthogonal two-dimensional subspaces $\mathcal{C} \oplus \mathcal{C}_1 \oplus \mathcal{C}_2 \oplus \mathcal{C}_3$ as shown in Figure 14.5. Since we have chosen a specific basis for each of these subspaces, we should give things a name.

We want to encode a single qubit, which lives in a two-dimensional space (spanned by $|0\rangle$ and $|1\rangle$), so it makes sense that we want our codespace to also be two-dimensional.

The (orthogonal) basis vectors of the codespace $\mathcal{C} = V_S$ are called **logical states**, and are usually taken to be the encodings of $|0\rangle$ and $|1\rangle$.

In general, the logical states will be *superpositions* of states, but we still sometimes refer to them as codewords.

In our example of the three-qubit code, we have the two logical states **logical 0** and **logical 1**, which we denote by

$$\begin{aligned} |0\rangle_L &:= |000\rangle \\ |1\rangle_L &:= |111\rangle. \end{aligned}$$

The justification for these names is twofold: firstly, $|0\rangle_L$ is exactly the encoding of $|0\rangle$, the “actual” zero state; and secondly, this state $|0\rangle_L$ will behave exactly as the zero state should when acted upon by the logical operators. For example, the operator X sends $|0\rangle$ to $|1\rangle$, so the *logical X* should send the *logical 0* to the *logical 1*. Let’s make this happen!

The normaliser of \mathcal{S} inside \mathcal{P}_3 is

$$N(\mathcal{S}) = \{1, XXX, -YYY, ZZZ\} \times \mathcal{S}$$

which we have written in such a way that we can just read off the cosets: there are four of them, and they are represented by 1 , XXX , $-YYY$, and ZZZ . These four (implementations of) logical operators all get given the obvious names:

$$\begin{aligned} 1_L &:= 1 \\ X_L &:= XXX \\ Y_L &:= -YYY \\ Z_L &:= ZZZ \end{aligned}$$

Note that the logical states for the three-qubit code are actually *not* superpositions. This reflects the fact that this code is really just a classical repetition code — it only protects against one type of error — embedded into the quantum world.

But note that these are not necessarily the smallest weight implementations! For example, any single Z_i (i.e. a Z acting on the i -th qubit) will have the same logical effect as ZZZ , as we can see by looking at how it acts on the logical states:

$$\begin{aligned} Z_1|1\rangle_L &= Z\mathbf{11}|111\rangle \\ &= -|111\rangle \\ &= ZZZ|111\rangle \\ &= ZZZ|1\rangle_L. \end{aligned}$$

In contrast, XXX is the smallest weight logical X implementation. The natural question to ask is then how to find all the implementations, but this is answered by going back to the very definition of them as coset representatives: *if P is some implementation of a logical operator, then so too is SP for any $S \in \mathcal{S}$* . In the example above, we see that $Z_1 = Z\mathbf{11}$ is exactly $\mathbf{1}ZZ \cdot ZZZ$. Because of this, we should really write something like $Z_L = ZZZS$, or $Z_L = [ZZZ]$, to make clear that ZZZ is just one specific representation of Z_L , but you will find that people often conflate implementations with the logical operators themselves and simply write $Z_L = ZZZ$.

Generally, for any CSS code encoding a single qubit into n -qubits, we define the logical X and logical Z operators to be the (equivalence classes of) the tensor products of all X operators or all Z operators (respectively), i.e.

$$\begin{aligned} X_L &:= X^{\otimes n} \\ Z_L &:= Z^{\otimes n}. \end{aligned}$$

Even more generally, for any $[[n, k, d]]$ code constructed from a stabiliser \mathcal{S} , it will be the case that $N(\mathcal{S})/\mathcal{S} \cong \mathcal{P}_k$.

Just a warning before moving on: this discussion might make logical states sound pointlessly simple — logical 0 is just given by three copies of $|0\rangle$, so what's the point? But this apparent simplicity is due to the fact that the three-qubit code is somehow not very quantum at all (these logical states are not superpositions), and in general things get a lot more complicated. For example, even with the three-qubit code, we shall see in Section ?? that

$$|+\rangle_L \neq |+++ \rangle$$

where, as per usual, $|+\rangle = H|0\rangle = (|0\rangle + |1\rangle)/2$.

14.5 ... and error families

The quotient group $N(\mathcal{S})/\mathcal{S}$ gave us logical operators, so the next thing to ask is what we get from the quotient group $\mathcal{P}_n/N(\mathcal{S})$.

Proving this is a bit of a task!

Recall that the elements of the quotient group G/H are exactly the cosets of $H \triangleleft G$.

The cosets of $N(\mathcal{S})$ inside \mathcal{P}_n are **error families** on the codespace $V_{\mathcal{S}}$. The individual elements of any error family (i.e. the elements of \mathcal{P}_n) are called **physical errors**.

Again, we can write \mathcal{P}_3 in such a way that we can immediately read off the cosets:

$$\mathcal{P}_3 = \{\mathbf{1}, X\mathbf{11}, \mathbf{1}X\mathbf{1}, \mathbf{11}X\} \times N(\mathcal{S}) \times \{\pm 1, \pm i\}.$$

Ignoring the phases, the three (non-trivial) error families are single bit-flips: $[X\mathbf{11}]$, $[\mathbf{1}X\mathbf{1}]$, and $[\mathbf{11}X]$; these error families X_i map the codespace \mathcal{C} to the subspace \mathcal{C}_i , as shown in Figure 14.6.

We sometimes denote the coset $P \cdot N(\mathcal{S})$ simply by $[P]$, just to save space.

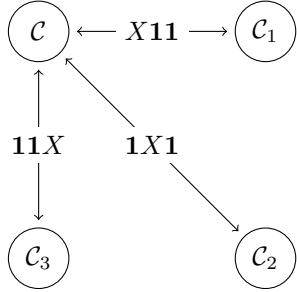


Figure 14.6: The single bit-flip error family X_i maps the codespace \mathcal{C} to the subspace \mathcal{C}_i , e.g. $X_2|000\rangle = |010\rangle \in \mathcal{C}_2$.

These errors also let us understand how the structure of the codespace is mirrored across each of the cosets. In other words, we picked \mathcal{C} to be our codespace, but what if we had instead picked \mathcal{C}_1 ? Well, we would get exactly the same code, just expressed in a different way, and this “different way” is described entirely by the error family $[X11]$. What we mean by this is the following:

- We can write \mathcal{C}_1 as the stabiliser space of \mathcal{S} conjugated by $X11$, i.e.

$$\begin{aligned} (X11)\langle ZZ1, 1ZZ \rangle (X11)^{-1} &= \langle (X11)(ZZ1)(X11)^{-1}, (X11)(1ZZ)(X11)^{-1} \rangle \\ &= \langle -ZZ1, 1ZZ \rangle \end{aligned}$$

and, indeed, $|100\rangle$ and $|011\rangle$ are both stabilised by this group.

- The logical states of \mathcal{C}_1 are, by definition as our chosen basis, the elements $|100\rangle$ and $|011\rangle$, but note that these are exactly the images of the logical states of \mathcal{C} under the error $X11$, i.e.

$$\begin{aligned} |0\rangle_{L,1} &:= |100\rangle = X11|000\rangle \\ |1\rangle_{L,1} &:= |011\rangle = X11|111\rangle \end{aligned}$$

- The logical operators on \mathcal{C}_1 are the logical operators on \mathcal{C} conjugated by $X11$, i.e.

$$\begin{aligned} X_{L,1} &:= (X11)(XXX)(X11)^{-1} \\ &= XXX \\ Z_{L,1} &:= (X11)(ZZZ)(X11)^{-1} \\ &= -ZZZ \end{aligned}$$

and, indeed, $X_{L,1}$ and $Z_{L,1}$ behave as expected on the new logical states, i.e.

$$\begin{aligned} X_{L,1}: |0\rangle_{L,1} &\longmapsto |1\rangle_{L,1} \\ |1\rangle_{L,1} &\longmapsto |0\rangle_{L,1} \\ Z_{L,1}: |0\rangle_{L,1} &\longmapsto |0\rangle_{L,1} \\ |1\rangle_{L,1} &\longmapsto -|1\rangle_{L,1} \end{aligned}$$

as you can check by hand.

All in all, the chain of normal subgroups

$$\mathcal{S} \triangleleft N(\mathcal{S}) \triangleleft \mathcal{P}_n$$

really does describe the full structure of the code: logical states, logical operators, and error families.

Recall that conjugation expresses a change of basis: given an invertible $(n \times n)$ matrix B , we can turn a basis $\{v_1, \dots, v_n\}$ into a new basis $\{Bv_1, \dots, Bv_n\}$, and to write any operator A in this new basis we simply calculate BAB^{-1} (“undo the change of basis, apply A , then redo the change of basis”).

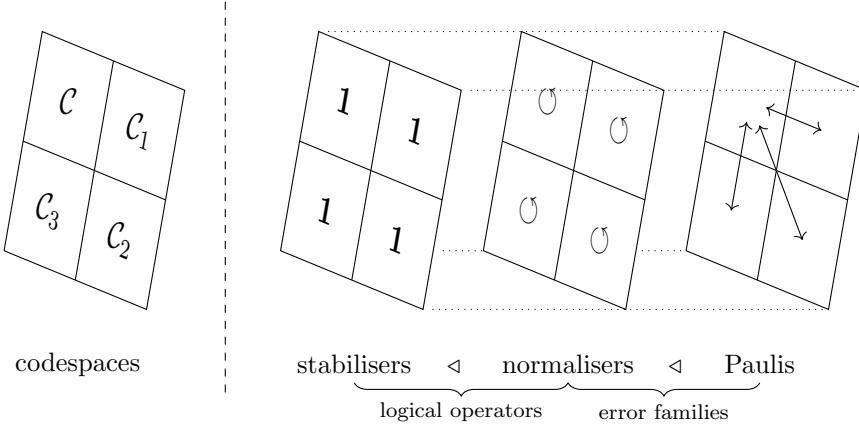


Figure 14.7: A visualisation of how the stabilisers, normalisers, and arbitrary Pauli operators act on the codespace decomposition: stabilisers act as the identity, normalisers move each subspace around within itself, and Pauli operators swap subspaces around between one another.

But this stabiliser formalism introduces some new ambiguity. In Section 14.3, we saw how measuring the three ancilla qubits in the Hamming [7, 4, 3] code gave us an error syndrome that we could use to determine on which qubit a Z -error had occurred, and back in Section 13.7 we saw the analogous error-syndrome setup for the three-qubit code. However, the stabiliser formalism is much more general: it makes no assumptions that only single-qubit errors can occur. This means that error syndromes will now only tell us which error *family* has occurred, not which specific *physical* error like they did before. At first, this seems like a definite downgrade from our previous theory — the actual errors that affect our circuits are still the *physical* errors, but now we have no way of knowing which one occurred, only which family it lives in! How are we to pick which coset representative to apply in order to correct the error?

As you might expect, the story is not yet over. Depending on the specifics of the scenario, sometimes knowing the error family is enough to be able to correct not just one physical error, but *many*. In order to give a more precise explanation, we need to take a step back and look at the scenarios that we’re actually trying to model — we do this in Section 14.7.

14.6 Logical operators (a different approach)

We have said a few times now that the main challenge in finding good quantum error-correction codes often lies in finding “good commuting sets of stabilisers”, so let’s take this seriously and try to rediscover the definitions from Sections 14.4 and 14.5 by starting with just commutativity.

Again, we already know that the Pauli matrices provide a useful basis with respect to which we can decompose the effects of any quantum channel, so we should carefully understand how the Pauli operators $P \in \mathcal{P}_n$ interact with any error-correcting code. For this, we introduce the notation

$$c(P, \sigma) := \begin{cases} 0 & P \text{ and } \sigma \text{ commute} \\ 1 & P \text{ and } \sigma \text{ anticommute} \end{cases}$$

for any Pauli operator P and any other operator σ . A particularly nice thing about this choice of definition (as opposed to taking $c(P, \sigma) \in \{\pm 1\}$, say) is that we can write

$$P\sigma = (-1)^{c(P, \sigma)} \sigma P.$$

“Correct the Paulis and you correct them all.”

Furthermore, this function has a nice relation on products: writing \oplus to mean addition mod 2, we can see that

$$c(P, \sigma\tau) = c(P, \sigma) \oplus c(P, \tau)$$

which reminds us of the fact that two anticommuting operators multiplied together produce a commuting operator.

Now fix some stabiliser group $\mathcal{S} = \langle g_1, \dots, g_{n-k} \rangle$. We define the **error syndrome** \underline{e}_P of a Pauli operator P to be the vector of all the values $c(P, g_i)$, i.e.

$$\underline{e}_P = (c(P, g_1), \dots, c(P, g_{n-k})).$$

It follows from the the above relation of how $c(P, -)$ turns products into sums that

$$\underline{e}_{P\sigma} = \underline{e}_P \oplus \underline{e}_\sigma.$$

The set of Pauli operators that have zero syndrome are special, and form a set known as the **normaliser**:

$$N(\mathcal{S}) := \{P \in \mathcal{P}_n \mid c(P, \sigma) = 0 \text{ for all } \sigma \in \mathcal{S}\}.$$

Since all elements of the stabiliser \mathcal{S} commute with one another, we know that $\mathcal{S} \subseteq N(\mathcal{S})$, but in general the normaliser is strictly larger. Now, by the definition of the normaliser and the multiplicative property of $c(P, -)$, if some Pauli operator P has a particular error syndrome \underline{e}_P then $P\sigma$ has the same error syndrome for any $\sigma \in N(\mathcal{S})$. This lets us gather together the Pauli operators into sets, which we call the **error cosets**, consisting of those Pauli operators which all have the same error syndrome. By the above, these can be described by some representative operator P , along with all other $P\sigma$ for $\sigma \in N(\mathcal{S})$, since the only way for P and Q to have the same error syndrome is for them to satisfy $Q = P\sigma$ for some $\sigma \in N(\mathcal{S})$.

Now for some counting. Since an error syndrome is exactly an n -bit string, there are 2^{n-k} possible different error syndromes. Each error coset is, by construction, of size $|N(\mathcal{S})|$. All together, the error cosets contain every single Pauli operator, of which there are 4^n . With this, we can calculate the size of the normaliser:

$$N(\mathcal{S}) = 4^n / 2^{n-k} = 2^{n+k}.$$

So the Pauli group is subdivided into error cosets by the normaliser, and every Pauli in the same coset has the same error syndrome. If we perform a syndrome measurement after passing through some noisy channel and get the result \underline{e} , then the effect of the channel is collapsed to being a linear combination of the terms inside the error coset corresponding to the error syndrome \underline{e} . By applying any element of that error coset, we are mapped back to the normaliser.

In fact, there is further substructure within the normaliser, and this is also reflected in each error family. Given a Pauli operator $P \in N(\mathcal{S})$ in the normaliser, we define its **logical syndrome** to be the vector $\underline{\ell}_P$ of all the values $c(P, \sigma)$, where σ ranges over $N(\mathcal{S})$. Note that, for any $\tau \in \mathcal{S}$, we have $\underline{\ell}_P = \underline{\ell}_{P\tau}$. Again, this splits the normaliser Pauli operators into sets, which we call the **logical cosets**, each being defined by having the same logical syndrome.

The error cosets divide up the Pauli group, with the normaliser as one specific example; the logical cosets divide up the normaliser, with the stabiliser as one specific example — see Figures 7.2 and 14.7.

Each operator that's in the normaliser but not in the stabiliser preserves the codespace (because it doesn't change the error syndrome), but it must do something non-trivial inside the codespace (because it's not in the stabiliser). It thus acts on the logical,

We saw in Section 14.3 that an $[n, k, d]$ code had $n - k$ stabiliser generators, so we preemptively label our generators from 1 to $n - k$.

encoded, qubits, and so we call it a **logical operator**. Moreover, these logical operators either commute or anticommute with one another. This should remind you of the Pauli operators themselves, and, indeed, we choose to associate these logical operators with **logical Pauli operators**. Which are which? Well, the choice is still arbitrary, as long as we get the relative commutation properties correct: it should be the case that $c(Z_L, X_L) = 1$, for example. In the case of a CSS code, there are always Z -type representatives that we can choose to take as the logical Z , and X -type representatives for the logical X .

Let's do some more counting. Since each logical coset is of size $|\mathcal{S}| = 2^{n-k}$, there must be $|N(\mathcal{S})|/|\mathcal{S}| = 4^k$ logical cosets, each corresponding to one of the 4^k logical Pauli operators on k qubits, and each described uniquely by a logical syndrome vector $\underline{\ell}$, and in such a way that every possible value of $\underline{\ell}$ is accounted for. However, recall that $\underline{\ell}_P = \underline{\ell}_{P\tau}$ for any $\tau \in \mathcal{S}$. This means that we don't need to record *all* the commutation values, but only a set of $2k$ many values, so that $\underline{\ell} \in \{0, 1\}^{2k}$. All in all, we can choose any linearly independent set of values we want for the generators, as long as we recall that any operator will commute with itself, and we require the symmetry $c(\sigma, \tau) = c(\tau, \sigma)$. For example, we could take

$$\begin{bmatrix} \underline{\ell}_1 \\ \underline{\ell}_2 \\ \vdots \\ \underline{\ell}_{2k} \end{bmatrix} = \begin{bmatrix} 0 & 1 \\ 1 & 0 \end{bmatrix}^{\oplus k}$$

which would naturally select pairs $(\underline{\ell}_{2n-1}, \underline{\ell}_{2n})$ as having the correct commutation relations necessary for them to act as logical Z and logical X for the n -th logical qubit.

Logical operators specify how to split the codespace, and are representatives of the logical cosets. In particular, the ± 1 eigenstates of Z_L define the **logical codewords**. Generically, these codewords are superpositions of basis states.

We emphasise out one final important point before returning to the example of the three-qubit repetition code.

While we *can* measure the *error* syndrome (all the stabilisers commute), we *cannot* measure the *logical* syndrome (not all the logical operators commute). Indeed, we must not even try — measuring just one such value is equivalent to performing a measurement of the logical qubit, destroying the superposition of the very state with which we're trying to compute!

So, back to the example of the three-qubit code from Section 13.1. Recall that the stabilisers are generated by $ZZ1$ and $1ZZ$. The normaliser is the set of Pauli operators that commute with all these stabilisers, which we can succinctly write as

$$N(\mathcal{S}) = \langle ZZ1, 1ZZ \rangle \times \{1, XXX, YYY, ZZZ\}$$

which already depicts the structure of the logical cosets: they are represented by XXX , YYY , and ZZZ . Using these representatives, we can evaluate the commutation properties:

$c(-, -)$	XXX	YYY	ZZZ
XXX	0	1	1
YYY	1	0	1
ZZZ	1	1	0

Here we're going to repeat some things that we already said in Sections 14.4 and 14.5.

From this we can see that *any* pair of these will work as the logical generators Z_L and X_L , since they all satisfy the required property of $c(Z_L, X_L) = 1$ and $c(Z_L, Z_L) = c(X_L, X_L) = 0$. In other words, although it's "natural" to define $Z_L := ZZZ$ and $X_L := XXX$, we could just as well decide to set $Z_L := XXX$ and $X_L = YYY$!

14.7 Error-correcting conditions

We can summarise the notion of a stabiliser code that we have defined rather succinctly: everything is determined by picking a stabiliser group, i.e. an abelian subgroup \mathcal{S} of the Pauli group \mathcal{P}_n that does not contain -1 . From this, we define the codespace to be the stabiliser subspace $V_{\mathcal{S}}$, the codewords to be a choice of basis vectors, the logical operators to be the cosets of $\mathcal{S} \triangleleft N(\mathcal{S})$, and the error families to be the cosets of $N(\mathcal{S}) \triangleleft \mathcal{P}_n$.

By setting up some ancilla qubits and constructing appropriate quantum circuits, we can enact any logical operator in such a way that we also measure an error syndrome, which points at a specific error family. But unlike in our study of the Steane code in Section 14.3, we can no longer simply apply the corresponding operator to fix the error, because the error is a whole coset — it contains many individual Pauli operators.

To fix an example to keep in mind, we return yet again to the three-qubit code. In Figure 14.8 we draw a diagram grouping together all the elements of \mathcal{P}_3 into the coset structure induced by $\mathcal{S} = \langle ZZ1, 1ZZ \rangle$. This is analogous to the diagrams that we saw back in Exercise 7.8.2, but with the simplification of ignoring phase.

$(11X) \cdot N(\mathcal{S})$	ZZX	$11Y$	$Z1X$	$1ZX$
	YYZ	XXZ	YXZ	$XY1$
	$XX1$	$YY1$	XYZ	YXZ
	$11X$	ZZX	$1ZY$	$Z1Y$
$(1X1) \cdot N(\mathcal{S})$	ZYZ	$1XZ$	$ZX1$	$1Y1$
	YZY	$X1Y$	$Y1X$	XZX
	$X1X$	YZX	XZY	$Y1Y$
	$1X1$	$ZY1$	$1YZ$	ZXZ
$(X11) \cdot N(\mathcal{S})$	YZZ	$X1Z$	$Y11$	$XZ1$
	ZYY	$1XY$	ZXX	$1YX$
	$1XX$	ZYX	$1YY$	ZXY
	$X11$	$YZ1$	XZZ	$Y1Z$
$N(\mathcal{S})$	ZZZ	$11Z$	$Z11$	$1Z1$
	YYY	XXY	YXX	XYX
	XXX	YYX	XYY	YXY
	1	$ZZ1$	$1ZZ$	$Z1Z$

$\leftarrow (ZZZ) \cdot \mathcal{S}$
 $\leftarrow (YYY) \cdot \mathcal{S}$
 $\leftarrow (XXX) \cdot \mathcal{S}$
 $\leftarrow \mathcal{S}$

We will see these circuits soon, starting in Section 14.9.

Formally, we can think of ignoring phase as looking at the quotient of \mathcal{P}_3 by the subgroup $\langle \pm 1, \pm i \rangle$, which results in an abelian group.

Figure 14.8: The entire group \mathcal{P}_3 with the coset structure induced by the stabiliser group $\mathcal{S} = \langle ZZ1, 1ZZ \rangle$. Note that we are ignoring global phase.

As we can see by looking at Figure 14.8, if we somehow measure an error syndrome pointing to the error family $[X11]$, for example, then there are 16 possible errors that could have occurred! We said that the stabiliser formalism would be *better* than our previous approach, so why do things seem so much worse now? Well, we are forgetting one key assumption that we made before that we have yet to impose in the stabiliser formalism: *up until now, we have only studied single-qubit errors*. Thinking back to our introduction of the three-qubit code in Section 13.1, we were specifically trying to deal with single bit-flip errors, i.e. only $X11$, $1X1$, and $11X$ (as well as the trivial error 111 , which we must not forget about, as we shall see). If we look back

at Figure 14.8 with this in mind, we notice something particularly nice: each of these single X -type errors lives in a different error family, and each error family contains exactly one of these errors.

In other words, *if we assume that only single bit-flip errors can occur*, then the stabiliser formalism describes errors in exactly the same way as before, since the error families are in bijection with the physical errors. But here is where the power of the stabiliser formalism can really shine through, since it allows us to understand what type of error scenarios our code can actually deal with in full generality. That is, rather than thinking about a code as something being built to correct for a specific set of errors, the stabiliser formalism lets us say “here is a code”, and *then* ask “for which sets of errors is this code actually useful?”. The answer to this question lies in understanding how any set of physical errors is distributed across the error families, and we can draw even simpler versions of the diagram in Figure 14.8 to figure this out.

Returning to the scenario where we assume that only single bit-flip errors can occur, we can mark the corresponding physical errors in Figure 14.8 — namely 1, $X11$, $1X1$, and $11X$ — with a dot. We do this in Figure 14.9, which is the first of many more diagrams of this form, which we call **error-dot diagrams**. Although we are working with the specific example of the three-qubit code in mind, these diagrams are meant to be understood more generally as applying to any stabiliser code. As we shall soon see, we don’t really need to worry about making sure that we have the right number of rows in each small rectangle (i.e. the right number of cosets of \mathcal{S} inside $N(\mathcal{S})$), and in some sense we don’t even really need to worry about what the physical errors *are*.

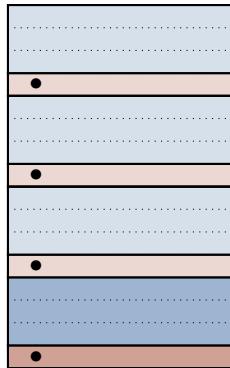
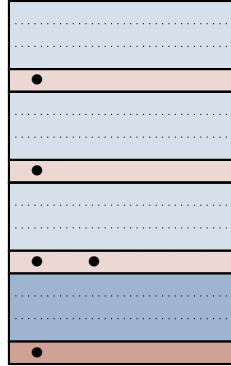


Figure 14.9: All specific X -type errors of weight at most 1 from Figure 14.8, each marked by a dot. The four cosets corresponding to $N(\mathcal{S}) \triangleleft \mathcal{P}_n$ are the error families, and we informally refer to the (copy of the) four cosets corresponding to $\mathcal{S} \triangleleft N(\mathcal{S})$ as **rows**.

As we said above, if each error family (i.e. coset) contains exactly one physical error (i.e. Pauli operator), then we already know how to apply corrections based on the error-syndrome measurements. In terms of the diagram in Figure 14.9, this rule becomes rather simple: *if each error family contains exactly one dot, then we can error correct*.

But can we say something more interesting than this? Well, let’s consider what happens if we have a diagram that looks like this:



That is, we're considering a scenario where there are *two* possible physical errors that can occur for a physical error syndrome. In the example of the three-qubit code, we're looking at the scenario where any single bit-flip error can occur, but *also* the operator $YZ1$ might affect our computation, enacting a bit-phase-flip on the first qubit and a phase-flip on the second. What would then happen if we measured the error syndrome $|01\rangle$? We know (from Section 13.7) that this corresponds to the error family $[X11]$, but both $X11$ and $YZ1$ live in this coset, so we're back to the question posed at the end of Section 14.5: how do we pick which operator to use to correct the error?

Here's the fantastic fact: in this case, *it doesn't matter!* Say we pick $X11$, but the physical error that had actually affected our qubits, originally in some encoded state $|\psi\rangle$, was $YZ1$. Then by applying the "correction" $X11$ our qubits would be in the state

$$(X11)(YZ1)|\psi\rangle = (ZZ1)|\psi\rangle$$

(where, once again, we ignore global phases). But $|\psi\rangle$ is, by construction, some codeword, which exactly means that it is stabilised by $ZZ1$, and so

$$(X11)(YZ1)|\psi\rangle = |\psi\rangle.$$

We can fully generalise this to improve upon the previous rule: *if all the dots in any given error family are all in the same row, then we can perfectly error correct.*

To prove this, we just return to the definition of cosets and the properties of the Pauli group. If two physical errors P_1 and P_2 are in the same row inside some family $E \cdot N(\mathcal{S})$, then by definition they both come from the same coset $P \cdot \mathcal{S}$, i.e.

$$\begin{aligned} P_1 &= EP'_1 \\ P_2 &= EP'_2 \end{aligned}$$

where $P'_1, P'_2 \in P \cdot \mathcal{S}$. Then EP corrects both P_1 and P_2 , since (again, we ignore global phase, which means that Pauli operators commute)

$$\begin{aligned} (EP)P_i &= (EP)(EP'_i) \\ &= E^2 PP'_i \\ &= PP'_i \in \mathcal{S} \end{aligned}$$

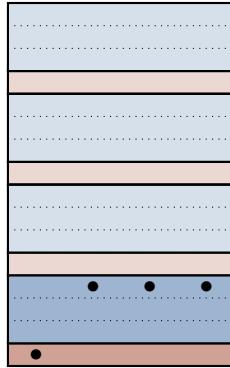
because Pauli operators square to 1, and $P'_i \in P \cdot \mathcal{S}$.

We also get the converse statement from this argument: *if any family contains dots in different rows, then we cannot error correct.* This is because we need EP to correct for some errors, and some different EP' to correct for others, and we have no way of choosing which one to correct with when we measure the error syndrome for E without already knowing which physical error took place.

So is this the whole story? Almost, but one detail is worth making explicit, concerning maybe the most innocuous looking error of all: the identity error family. Consider a scenario like the following:

This is one of those arguments where it's easy to get lost in the notation. Try picking two physical errors P_1 and P_2 in the same row somewhere in Figure 14.8 and following through the argument, figuring out what E , P , P'_1 , and P'_2 are as you go.

Just to be clear, if we knew which physical errors took place, then we wouldn't have to worry about error correction at all, because we'd always know how to perfectly recover the desired state. And remember that we can't measure to find out which physical error took place, since this would destroy the state that we're trying so hard to preserve!



In the case of the three-qubit code, this corresponds to the possible physical errors being single phase-flips $Z11$, $1Z1$, and $11Z$. But here we see how misleading it is to omit mention of the identity error 111 , because the single phase-flips all live in the same $N(\mathcal{S})$ coset as 111 , but different \mathcal{S} cosets. That is, they are in the same error family, but a different row. By our above discussion, this means that we cannot correct for these errors — indeed, if we measure the error syndrome corresponding to “no error”, then we don’t know whether there truly was no error or if one of these single phase-flips happened instead. To put it succinctly, we nearly *always* make the assumption that *no errors at all might occur*, which is exactly the same as saying that *the trivial error 1 might occur*. This means that we cannot correct for any errors that are found in the normaliser of \mathcal{S} but not in \mathcal{S} itself. Although this is technically a sub-rule of the previous rule, it’s worth pointing out explicitly.

An error-dot diagram describes a perfectly correctable set of errors if and only if the following two rules are satisfied:

1. In any given error family, all the dots are in the same row.
2. Any dots in the bottom error family are in the bottom row.

(The second rule follows from the first as long as the scenario in question allows the possibility for no errors to occur.)

Of course, we can state these conditions without making reference to the dot-error diagrams, instead using the same mathematical objects that we’ve been using all along. Proving the following version of the statement is the content of Exercise 14.11.12.

Let $\mathcal{E} \subseteq \mathcal{P}_n$ be a set of physical errors. Then the stabiliser code defined by \mathcal{S} can perfectly correct for all errors in \mathcal{E} if and only if

$$E_1^\dagger E_2 \in N(\mathcal{S}) \setminus \mathcal{S}$$

for all $E_1, E_2 \in \mathcal{E}$.

You might notice that we’ve been sometimes been saying “perfectly correctable” instead of just “correctable”. This is because there might be scenarios where we are happy with being able to correct errors not perfectly, but instead merely with some high probability.

These dot-error diagrams are also able to describe more probabilistic scenarios. We have been saying “single-qubit errors”, but we could just have well have been saying “lowest-weight errors”, and then the assumption that errors are independent of one another means that higher-weight errors happen with lower probability. But

the stabiliser formalism (and thus the error-dot diagrams) don't care about this "independent errors" assumption! What this means is that we could refine our diagrams: instead of merely drawing dots to denote which errors can occur, we could also label them with specific probabilities. So we could describe a scenario where, for example, one specific high-weight error happens annoyingly often.

One last point that is important for those who care about mathematical correctness concerns our treatment of global phases. We *do* need to care about global phases in order to perform error-syndrome measurements, but *once we have the error syndrome we can forget about them*. In other words, we need the global phase in order to pick the error family, but not to pick a representative within it.

14.8 Code distance and thresholds

Given an error model in which, in principle, all Pauli errors are possible but low-weight errors are more likely than the high-weight errors, it makes perfect sense to look for an error correcting a code which can perfectly correct errors with weight at most t for some "good" value of t . Such a code will fail will with probability roughly equal to the total probability of any error of weight larger than t occurring. This probability of failure is called the **logical error probability**. The goal of quantum error correction is to use all the tricks we have discussed so far (and many more) to realise *logical* qubits with logical error rates *below* the error rate of the constituent *physical* qubits.

As in the case of classical codes, the **distance** of a quantum code is defined as the minimum weight error that can go undetected by the code. In other words, it is the minimum weight Pauli operator than can transform one codeword state into another. But as we've seen, all such operators are in $N(\mathcal{S}) \setminus \mathcal{S}$, which means that

$$d = \min_{P \in N(\mathcal{S}) \setminus \mathcal{S}} |P|.$$

Now our goal is less ambitious: we are not aiming to correct *all* possible Pauli errors, but only those of weight at most t , where t satisfies $d = 2t + 1$. So how can a code with distance d do this?

Firstly, note that, if we take a product of two errors E_i and E_j , each of weight at most t , then the resulting Pauli operator $E_i E_j$ will have weight at most $2t$, and by definition $2t < d$. Therefore the product of these errors can never be a logical operator, since the logical operators in $N(\mathcal{S}) \setminus \mathcal{S}$ have weight at least d . Thus if one of these errors E_i occurs and our decoding procedure picks another error E_j that gives rise to the same syndrome (i.e. that belongs to the same error family) and applies the latter to the encoded qubits, then we know that $E_i E_j \notin N(\mathcal{S}) \setminus \mathcal{S}$, which means that $E_i E_j \in \mathcal{S}$ acts as the identity on the codespace.

Needless to say, from the perspective of code distance alone, the larger the value of d the better we can correct for more errors. For this, we need the logical errors (i.e. the logical operations on the codespace $L \in N(\mathcal{S}) \setminus \mathcal{S}$) to have the largest possible weight — by our assumptions about our error model, these occur with low probability, and thus keep the logical error probability low.

The **threshold theorem** for stabiliser codes asserts that if the physical error probability p of individual qubits is below a certain threshold value p_{th} then increasing the distance of the code will decrease the logical error probability. This principle implies that quantum error-correction codes *could theoretically* suppress the logical error rate indefinitely. However, if the physical error rate p is *greater than* the threshold value p_{th} , then quantum encoding actually becomes counterproductive. So the threshold value serves as a critical experimental benchmark for quantum computing experiments, since achieving it is essential for the feasibility of quantum error correction. We will return to the threshold theorem in more detail in Chapter 15.

As of 2024, the upper bound for this threshold value is approximately $p_{\text{th}} = 0.1$.

We are being slightly informal with the way we draw these dot-error diagrams: cosets of \mathcal{S} itself inside \mathcal{P}_n don't make sense, as we've said, because \mathcal{S} is generally not normal inside \mathcal{P}_n . Also, when we quotient by $\{\pm 1, \pm i\}$ (by drawing just a single sheet, instead of four as in the diagrams in Exercise 7.8.2), we make \mathcal{P} abelian, and this makes the normaliser no longer the actual normaliser.

Recall that the weight $|P|$ of a Pauli operator $P = P_1 \otimes \dots \otimes P_n$ is the number of non-identity P_i . For example, **111** has weight 0, **Z11** and **1X1** have weight 1, and **XXX** has weight 3.

14.9 Encoding circuits

The previous sections have set up a lot of abstract theory about stabiliser codes, so now let's take some time to look at more concrete aspects, such as the quantum circuits that actually let us build these codes "in practice".

At the end of Section 14.3 we showed that the CSS construction could be applied to the Hamming [7, 4, 3] code over itself to obtain the so-called Steane [[7, 1, 3]] code, which has generators G_1, \dots, G_6 given by the rows in the matrix

$$\begin{bmatrix} X & X & 1 & X & X & 1 & 1 \\ X & 1 & X & X & 1 & X & 1 \\ 1 & X & X & X & 1 & 1 & X \\ Z & Z & 1 & Z & Z & 1 & 1 \\ Z & 1 & Z & Z & 1 & Z & 1 \\ 1 & Z & Z & Z & 1 & 1 & Z \end{bmatrix}$$

and codespace given by the corresponding stabiliser space $\mathcal{C} = V_S$, where $S = \langle G_1, \dots, G_6 \rangle$.

Now, what are the logical states for this code? Well, by definition they should be basis states for the stabiliser space $V_{\langle G_1, \dots, G_6 \rangle}$, but the "real" motivation for them is that they should just be the encodings of $|0\rangle$ and $|1\rangle$ in the code. So the question becomes just *how do we actually encode states with a code described by the stabiliser formalism?* But it turns out that we have already secretly answered this question in Exercise 7.8.5: the projector onto the ± 1 -eigenspace of any G_i is given by $\frac{1}{2}(\mathbf{1} \pm G_i)$.

Note that this matrix is just like two copies of the generator matrix for the Hamming [7, 4, 3] code stacked on top of one another: the first with X -type stabilisers, and the second with Z -type stabilisers.

Given stabiliser generators G_1, \dots, G_s , the projector onto the stabiliser space $V_{\langle G_1, \dots, G_s \rangle}$ (i.e. the encoding for the corresponding stabiliser code) is given by

$$\prod_{i=1}^s \frac{1}{2}(\mathbf{1} + G_i).$$

In other words, we want to define

$$|0\rangle_L := \frac{1}{2^6} \left(\prod_{i=1}^6 (\mathbf{1} + G_i) \right) |0\rangle^{\otimes 7}$$

since this will be in the $+1$ -eigenspace of all of the G_i , which is exactly the stabiliser space $V_{\langle G_1, \dots, G_6 \rangle}$. Similarly, we set

$$|1\rangle_L := \frac{1}{2^6} \left(\prod_{i=1}^6 (\mathbf{1} + G_i) \right) |1\rangle^{\otimes 7}.$$

One thing to note is that the order of the product over the G_i doesn't matter here: by design, every stabiliser generator commutes with every other, since they "overlap" (i.e. have non-identity terms) in an *even* number of positions, so any -1 signs arising from anticommutativity will cancel out with one another. So for $|0\rangle_L$, when we expand out the product $\prod_i (\mathbf{1} + G_i)$, we can simply move all the Z -type terms to the right and then forget them, since Z acts trivially on $|0\rangle$.

This means that we're left with only the X -type terms, and there are eight of these:

$$\begin{aligned} |0\rangle_L &:= \frac{1}{\sqrt{2^3}} (\mathbf{1} + G_1 + G_2 + G_3 \\ &\quad + G_1 G_2 + G_1 G_3 + G_2 G_3 + G_1 G_2 G_3) |0000000\rangle \\ &= \frac{1}{\sqrt{8}} (|0000000\rangle + |1101100\rangle + |1011010\rangle + |0111001\rangle \\ &\quad + |0110110\rangle + |1010101\rangle + |1100011\rangle + |0001111\rangle). \end{aligned}$$

This state is not normalised, since it's given by a bunch of projections one after the other, but we won't worry about this until we first make some simplifications.

We *need* all the generators to commute in order for the simultaneous $+1$ -eigenspace to exist!

If you look up the codewords for the Steane code elsewhere, you might find different expressions, but this is simply an artifact of expressing the parity check matrix of the Hamming code in a different basis. Note also that here we have normalised the state.

You can check by hand that this superposition is indeed invariant under each of the G_i . Now, we could perform a similar calculation for $|1\rangle_L$, but since we have a CSS code we already know that $X_L = X^{\otimes 7}$ is an implementation for the logical X operator, so we can simply use this:

$$\begin{aligned} |1\rangle_L &:= X_L|0\rangle_L \\ &= \frac{1}{\sqrt{8}}(|1111111\rangle + |0010011\rangle + |0100101\rangle + |1000110\rangle \\ &\quad + |1001001\rangle + |0101010\rangle + |0011100\rangle + |1110000\rangle). \end{aligned}$$

We know what the logical states are, and by the previous discussions we also know what the logical operators are: cosets of $\langle G_1, \dots, G_6 \rangle$ within its stabiliser in \mathcal{P}_7 . For example, not only is $X^{\otimes 7}$ an implementation of X_L , but so too is $11X11XX$.

So how can we actually access these logical states in order to do computation with them? In other words, we need to design an **encoding circuit** that allows us to prepare the states $|0\rangle_L$ and $|1\rangle_L$ so that we can then perform computation on them. As above, we will be able to neglect the Z -type stabilisers, because we're working in the computational basis. More specifically, since $|0\rangle$ and $|1\rangle$ live in the ± 1 -eigenspace for Z , we don't need to further project them to the stabiliser spaces of the Z -type stabilisers; we start with a basis in the stabiliser space for $\pm Z$, and when we encode we obtain a basis in the stabiliser space for S and $\pm Z_L$. This sort of duality always happens for CSS codes, and note that the choice of X versus Z isn't "special" — if we switch to the $|\pm\rangle$ basis then it would suffice to measure Z -type stabilisers, since we are already in the ± 1 -eigenspace for X . If this seems confusing, then don't worry: look at the circuits below, follow the evolution of the input state through them, and then see what would happen if you did the same thing after adding the gates for the three missing Z -type stabilisers as well. You will see that (up to a possible global phase) nothing changes.

Inspired by the classical Hamming [7, 4, 3] code, we can think of the last three qubits in our seven-qubit encoding as the parity-check qubits, and read off the layout of the circuit from the parity-check matrix: the (3×3) identity submatrix corresponds to the controls. This gives us the encoding circuit in Figure 14.10.

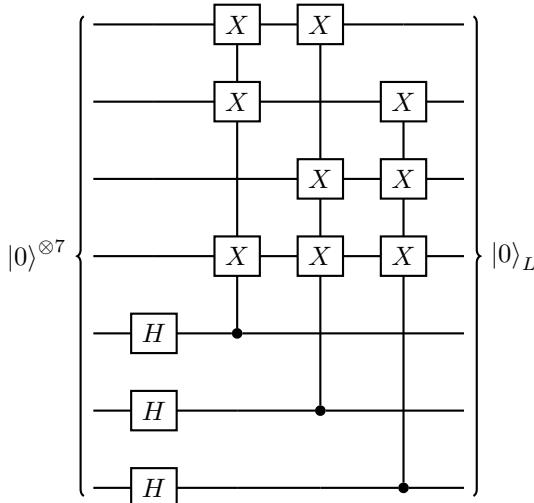
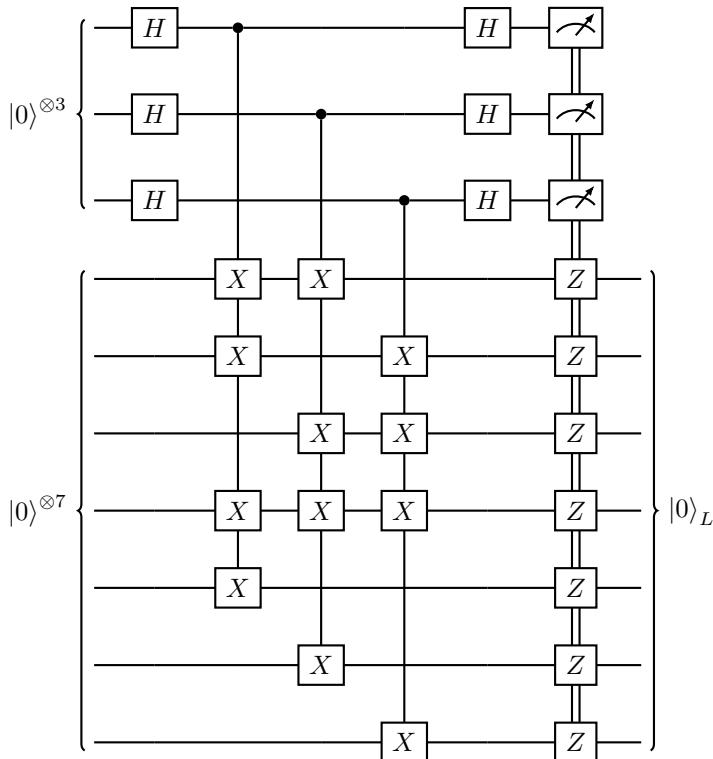


Figure 14.10: One possible encoding circuit for the Steane code, requiring no ancilla bits.

You can check this by hand: see that $11X11XX$ sends $|0\rangle_L$ to $|1\rangle_L$.

14.10 Encoding arbitrary states

The encoding circuit in Figure 14.10 describes a unitary operation (it has no measurements), and its particularly compact form makes it very useful for certain complexity-theoretic calculations, but it has one major drawback: it is not itself protected against errors! If we are trying to design things for the real world, where qubits can undergo decoherence, then we should compensate for this in *all* our quantum computation, *including* the circuits we use to prepare states. We have already done the hard work for this though, in Section 7.4, when we constructed circuits to project onto Pauli stabiliser spaces. This gives us the encoding circuit in Figure 14.11.



If you want people to be able to stay dry if it's raining, then you might build a tunnel from location A to location B so that they can use this for cover. But this isn't going to stop people from getting wet on their (necessary) journey from their home to location A!

Figure 14.11: Another possible encoding circuit for the Steane code, which uses three ancilla bits for error correction when encoding arbitrary states, but is non-unitary (since it involves measurement). The measurements of the ancilla bits can be used to apply the necessary Z -type corrections.

The three measurements in the encoding circuit in Figure 14.11 allow us to correct for any single-qubit error in the encoding process, just as we did in Section 7.4, using the lookup table from Section 14.3. If we measure $(+++)$, then no error has occurred, but if we measure, say, $(- + +)$, then we know that the error Z_5 has affected our encoding, and so we must correct for this. Of course, as we now know from Section 14.7, what it means to correct for the Z_5 error depends on which errors can possibly occur. If we make the usual assumption that only errors of weight 1 (i.e. single-qubit errors) can occur, then the Z_5 error is exactly that: a phase-flip on the fifth qubit.

So now we have seen two circuits for encoding the logical 0 state, but what about if we want to encode an arbitrary state? That is, we already have some qubit in an interesting state $|\psi\rangle$ and we want to use the Steane code to protect it against decoherence.

Before we look at this question, it's important to mention something about practical use here. As is often the case, a chain is only as strong as its weakest link, and the process of encoding a single qubit into seven qubits is a particularly error-prone

What we say here can be applied to other stabiliser codes, but we stick with the Steane code to make it easier to look at specific examples.

process. In practice, it is much more desirable to start with logical 0 and *then* do all of our computation, knowing that we are already in the “protected” world of a stabiliser code.

We know that all the X -type stabilisers for the Steane code have an even number of X terms in them, and so will commute with any implementation of the logical X operator X_L . Since the (bottom register of the) encoding circuit in Figure 14.11 simply applies the X -type stabilisers to $|0\rangle^{\otimes 7}$, we can use this commutativity. Indeed, by construction of the logical operators and the logical states, we know that encoding $|0\rangle^{\otimes 7}$ to $|0\rangle_L$ and then applying X_L gives us the state $|1\rangle_L$. But then the commutativity of X_L with the X -type stabilisers tells us that we *also* obtain $|1\rangle_L$ if we first apply X_L to $|0\rangle^{\otimes 7}$ and then encode. Symbolically, writing E to mean the operation of applying the encoding circuit,

$$\begin{aligned}|1\rangle_L &= X_L|0\rangle_L \\ &= X_L E |0\rangle^{\otimes 7} \\ &= E X_L |0\rangle^{\otimes 7}\end{aligned}$$

where we can pick any implementation of X_L that we like, such as $X^{\otimes 7}$ or **11X11XX**. This tells us that there are two ways of obtaining $|1\rangle_L$ from $|0\rangle_L$:

1. apply X_L and then encode
2. encode and then apply X_L .

Now let’s generalise this, replacing the X_L with a controlled version, controlled exactly by the state $|\psi\rangle$ that we wish to encode. If $|\psi\rangle = \alpha|0\rangle + \beta|1\rangle$, then we want to construct the logical state

$$|\psi\rangle_L := \alpha|0\rangle_L + \beta|1\rangle_L.$$

Let’s look at the first option from above: applying X_L and then encoding. For a simpler circuit, we can use the low-weight implementation **11X11XX** of X_L , so that we prepare the state

$$\alpha|0\rangle_L + \beta|1\rangle_L = \alpha|0000000\rangle + \beta|0010011\rangle$$

and then feed this into the encoding circuit from before. This gives us the circuit in Figure 14.12.

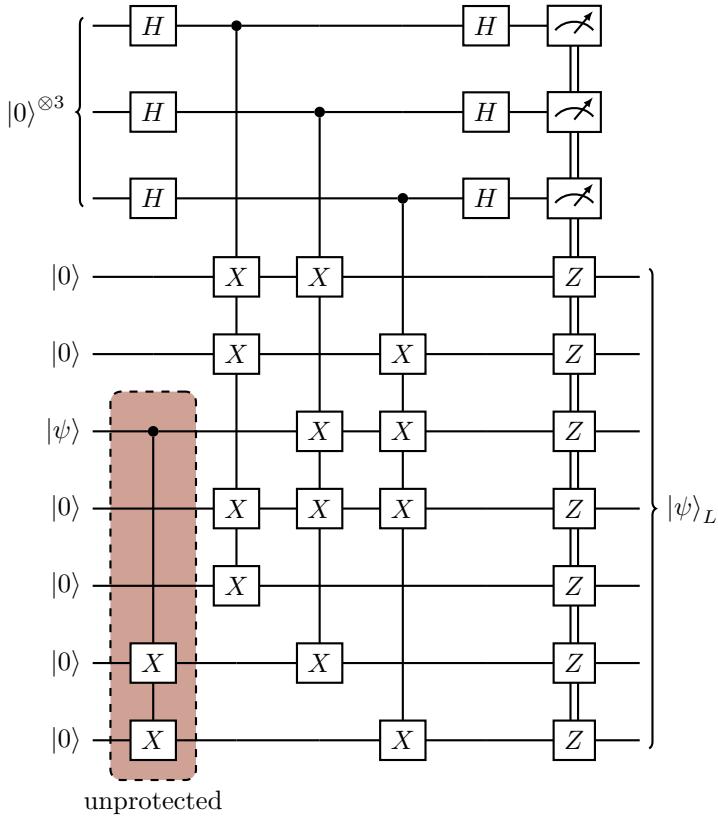


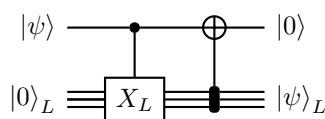
Figure 14.12: Preparing the logical version $|\psi\rangle_L$ of an arbitrary state $|\psi\rangle$ in a way that allows us to correct for any single-qubit errors in the *encoding* process, but *not* the *preparation* process (highlighted in red).

To repeat ourselves, the very first step of this circuit that enacts

$$|0\rangle|0\rangle|\psi\rangle|0\rangle|0\rangle|0\rangle \mapsto \alpha|0000000\rangle + \beta|0010011\rangle$$

(where $|\psi\rangle = \alpha|0\rangle + \beta|1\rangle$) is *not* protected by any error correction scheme. If $|\psi\rangle$ is some easily reproducible state, like $|0\rangle$, then we don't really mind so much, since we could instead use a circuit where all seven qubits are initially in state $|\psi\rangle$, avoiding this problem altogether. But if $|\psi\rangle$ is the outcome of some previous computation, or just a state that we don't have complete knowledge of, then we will always be faced with some uncertainty — did this preparation part of the circuit undergo an error or not? These sorts of problems are avoided if we can design truly fault-tolerant computational systems, instead of relying on mere error correction.

Now let's look at the second option from before: encoding and then applying X_L . Imagine that we were able to construct the following circuit:



This would transfer the state $|\psi\rangle = \alpha|0\rangle + \beta|1\rangle$ into a logical version $|\psi\rangle_L$, since it enacts the transformations

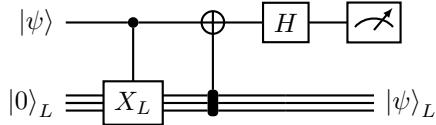
$$\begin{aligned} (\alpha|0\rangle + \beta|1\rangle)|0\rangle_L &\mapsto \alpha|0\rangle|0\rangle_L + \beta|1\rangle|1\rangle_L \\ &\mapsto |0\rangle(\alpha|0\rangle_L + \beta|1\rangle_L). \end{aligned}$$

But what do we actually mean by this circuit? We haven't defined controlled- X_L , nor what it means for a c-NOT to be controlled by a logical state.

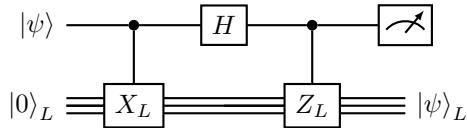
Thanks to the no-cloning theorem (Section 5.9), we know that there is no way of getting around this problem of only having one copy of the state $|\psi\rangle$ that will work for any possible input — only if $|\psi\rangle$ is something already known. So the preparation part of the circuit doesn't clone the input state, but instead "smears it out" across three qubits instead of one, just like we mentioned in Section 13.7.

The first is reasonably simple: if the control qubit is in state $|1\rangle$, then we want to apply X_L to the target. Since X_L can be expressed as a tensor product of Pauli X operators, this means that the controlled- X_L is just a bunch of controlled-NOT gates, each controlled by the top qubit, and targeting each of the qubits of the encoded state.

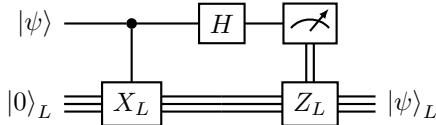
The second step is maybe not so obvious, but there's a trick that we can use here! We know that the top qubit should end in the $|0\rangle$ state, so we can do anything we want to it. For example, let's apply a Hadamard gate and then measure it — why not?



But now we can recall how controlled-NOT (which is simply a controlled- X) interacts with the Hadamard: the circuit above is equivalent to the circuit below.



This is a circuit that we could build, since we know all about the many implementations of X_L and Z_L thanks to the stabiliser formalism. If we really like, however, we could go one step further and replace the controlled- Z_L with a Z_L after the measurement:



14.11 Remarks and exercises

14.11.1 Error correcting conditions for the three-qubit code

When building the Shor $[[9, 1, 3]]$ code, we used the three-qubit code twice: once to correct for X -errors, and once for Z -errors. However, there is no reason why we couldn't instead have used one copy to correct for Y -errors instead of X -errors, and we can see this using the dot-error diagrams.

Consider the stabiliser code given by $S = \langle ZZ1, 1ZZ \rangle$. In Figure 14.13 we draw all the computational errors of weight at most 1, and we see that this does *not* describe a correctable scenario. That is, as we already know, the three-qubit code cannot alone correct for all single-qubit errors.

If we want to keep the circuit as simple as possible, then we should choose the smallest weight representative of X_L , which might not be just a tensor product of all X operators. For example, in the seven-qubit code there is an implementation of X_L of weight 3.

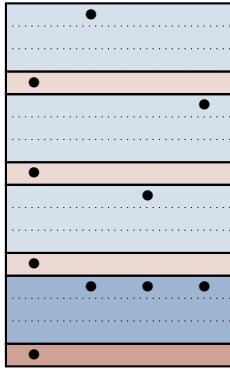


Figure 14.13: All computational errors of weight at most 1 for the three-qubit code given by $\langle ZZ1, 1ZZ \rangle$. Note that this describes a *non*-correctable scenario. In fact, *both* of the two rules are broken.

But now let's look at X -, Y -, and Z -errors all individually and see what happens. As Figure 14.14 shows, the three-qubit code given by $\langle ZZ1, 1ZZ \rangle$ can correct for all X -errors (as we already knew), but also for all Y -errors!

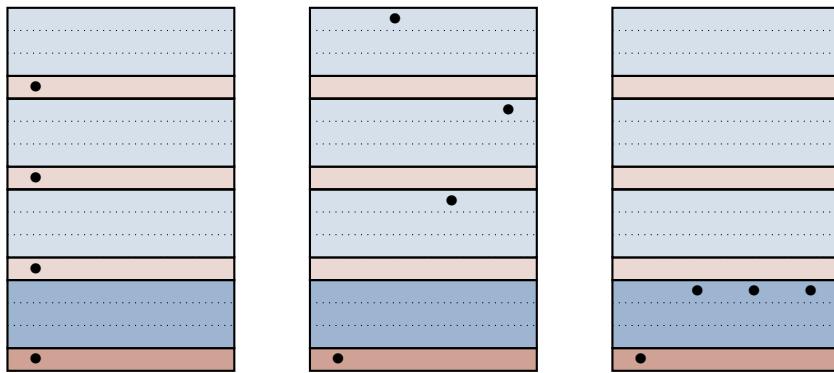


Figure 14.14: Different types of computational errors of weight at most 1. *Left to right:* X -type errors (correctable); Y -type errors (correctable); Z -type errors (non-correctable).

14.11.2 The smallest $d = 3$ code, full stop

We have already seen the Shor $[[9, 1, 3]]$ code, which can protect against any single-qubit error. Despite its simplicity, it is not the smallest code that can do this: that title belongs to the $[[5, 1, 3]]$ stabiliser code, given by

$$\mathcal{S} = \langle XZZX1, 1ZXXZ, X1XZZ, ZX1XZ \rangle$$

shown as a Tanner graph in Figure ???. Note that the stabiliser generators (i.e. parity checks) are related to each other by cyclic shifts: we just take different length-five chunks from the infinite string $XZZX1XZZX1XZZX1\dots$. This code is unusual compared to the ones we have seen before, since it's actually impossible to write its generators as operators that consist of either all X or all Z .

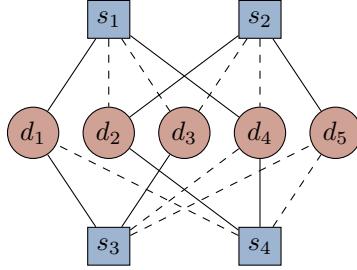


Figure 14.15: The Tanner graph for the $[[5, 1, 3]]$ stabiliser code. Solid lines represent X -checks, and dashed ones Z -checks.

This $[[5, 1, 3]]$ code truly is optimal, in that it is the smallest possible quantum error correcting code with $d = 3$. Indeed, suppose that we have n qubits representing one logical qubit, and each error X , Y , or Z on any of these n qubits maps the two-dimensional codespace to a different, mutually orthogonal subspace. This means that we have to fit the codespace, plus $3n$ two-dimensional subspaces, into the 2^n -dimensional Hilbert space associated with the n qubits. This implies that we need to satisfy

$$2(1 + 3n) \leq 2^n$$

which tells us that we must have at least $n \geq 5$.

The counting argument above tells us that the smallest code must have *at least* five qubits, but doesn't tell us if we can actually make one with exactly five qubits! How do we actually go about finding optimal codes then? The answer is simply that we do not know — there is no universal prescription for designing optimal quantum codes. But we do know quite a few things about designing good quantum codes.

One last thing to mention is how this code displays that quantum codes can be used for more than just error correction. The $[[5, 1, 3]]$ code gives a way of designing a [\(\(3, 5\)\) quantum secret sharing](#) protocol.

14.11.3 Hamming code encodings and decodings

1. How is the binary message 0101 encoded in the Hamming $[7, 4, 3]$ code?
2. If we receive the string 1011011 from Alice, who encoded her message in the Hamming $[7, 4, 3]$ code, then what is the error syndrome? What correction should we make? What is the decoded message?

14.11.4 Generator and parity-check matrices

Show that, if the $(n \times k)$ generator matrix for an $[n, k, d]$ linear code is written in the form

$$G = \begin{bmatrix} \mathbf{1} \\ P \end{bmatrix}$$

where P is an $((n - k) \times k)$ binary matrix, then the parity-check matrix can be written as

$$H = [P \quad \mathbf{1}] .$$

Note that this code is *not* a CSS code! To prove this, we could use theorems about **transversal gates**. The smallest CSS code with $d = 3$ is described in Exercise 14.11.10.

Here we are tacitly assuming that the code is **non-degenerate** (see Exercise 14.11.9).

14.11.5 A big parity check matrix

Consider the following parity-check matrix of a classical $[n, k, 7]$ code:

$$\left[\begin{array}{cccccccccccccccccccc} 1 & 0 & 0 & 1 & 1 & 1 & 0 & 0 & 0 & 1 & 1 & 1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 1 & 0 & 1 & 0 & 1 & 1 & 0 & 1 & 1 & 0 & 0 & 1 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 1 & 0 & 1 & 1 & 0 & 1 & 1 & 0 & 1 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 1 & 0 & 1 & 1 & 1 & 0 & 1 & 1 & 0 & 1 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 1 & 1 & 0 & 0 & 1 & 1 & 1 & 0 & 1 & 1 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \\ 1 & 1 & 0 & 1 & 0 & 1 & 1 & 1 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \\ 1 & 1 & 0 & 1 & 1 & 0 & 0 & 1 & 1 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \\ 1 & 1 & 1 & 0 & 0 & 1 & 0 & 1 & 0 & 1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \\ 1 & 1 & 1 & 1 & 0 & 0 & 0 & 0 & 1 & 1 & 1 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \\ 0 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \end{array} \right]$$

1. What are the values of the parameters n and k ?

2. If we receive the bit string

$$x = 00101001011011000110101$$

and assume that no more than three errors have occurred, what are the locations of the errors?

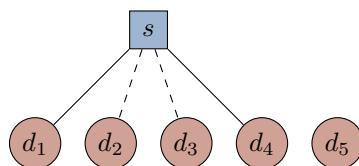
3. Show that we could use two copies of this code to build a CSS code.
4. If we build a CSS code using this classical code, what parameters does it have? That is, what is its specification as an $[[n, k, d]]$ code?
5. Given a state $|\psi\rangle$ of 23 qubits, how would you measure the value of the first stabiliser

$$X_1 X_4 X_5 X_6 X_{10} X_{11} X_{12} X_{13} ?$$

6. If we were to write out $|0\rangle_L$ for the CSS code, how many different basis states would be in the superposition?

14.11.6 Using Tanner graphs

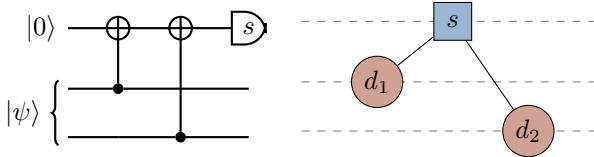
Consider the Tanner graph below.



Recall that we use solid lines to denote X -parity checks and dashed lines to denote Z -parity checks.

1. What stabiliser does this Tanner graph define?
2. Add to the Tanner graph the definition for a second stabiliser $g_2 = 1XZZX$. How can we visually confirm that the two stabilisers commute?

At the end of Section 14.1, in Figure 14.3, we claimed an equivalence between Tanner graphs (for detecting the parity of X errors) and circuits. Consider the simpler example below.



3. Prove that these two are equivalent.
4. Draw the circuit for measuring the parity of Z errors.
5. Draw the Tanner graph for the Shor $[[9, 1, 3]]$ code.

14.11.7 Five-qubit repetition code

Consider the five-qubit repetition code

$$\begin{aligned} |0\rangle &\mapsto |+\rangle^{\otimes 5} \\ |1\rangle &\mapsto |-\rangle^{\otimes 5}. \end{aligned}$$

1. What are the stabilisers of this code?
2. What is the normaliser of this code?
3. Which of the following sets of errors satisfy the error correcting conditions for this code? a. X_1, Z_5 b. X_1, X_2, X_3, X_4 c. Z_1, Z_2, Z_3, Z_4 d. Z_1Z_2, Z_2Z_4, Z_1Z_4

14.11.8 An error in the Steane $[[7, 1, 3]]$ code

One logical qubit is encoded in seven physical qubits using the Steane $[[7, 1, 3]]$ code, which then experiences the error

$$E = X1Y11Z1.$$

1. What is the resulting error syndrome?
2. What is the smallest-weight error with the same error syndrome as E ?
3. If we apply the smallest-weight error from above as our correction, then what is the net logical error on the encoded qubit?

14.11.9 Non-degenerate codes

An $[n, k, d]$ code is said to be **non-degenerate** if every Pauli operator of weight $\leq \lfloor d/2 \rfloor$ has a distinct error syndrome.

Prove that all the stabilisers for a non-degenerate code have weight $\geq d$.

14.11.10 The smallest $d = 3$ CSS code

The $[[7, 1, 3]]$ code is the smallest possible CSS code with distance $d = 3$. Let's prove that now, making the simplification that we will only consider *non-degenerate* codes.

1. If we construct a CSS code using two parity-check matrices H_1 and H_2 , with m_1 and m_2 rows (respectively), and we want our code to encode one logical qubit into n physical qubits, then how are the numbers m_1, m_2 , and n related?
2. Explain why the columns of a non-degenerate $d = 3$ code must be distinct. Hence conclude that $2^{m_i} \geq n + 1$ for $i = 1, 2$.
3. Conclude that the smallest possible non-degenerate CSS code with $d = 3$ has $n = 7$ qubits.

14.11.11 CSS codes from a single matrix

Let H be an $(n \times m)$ binary matrix, with $m > n$, whose rows are linearly independent. When taken as a parity-check matrix, it thus defines an $[m, m - n, d]$ code. Even though H^T cannot be the parity-check matrix of a code (simply because $m > n$), it still has a well defined distance d^T .

Show that

$$\begin{aligned} H_X &= [\mathbf{1}_{m \times m} \otimes H \quad H^T \otimes \mathbf{1}_{n \times n}] \\ H_Z &= [H \otimes \mathbf{1}_{n \times n} \quad \mathbf{1}_{m \times m} \otimes H^T] \end{aligned}$$

defines a CSS code with specification $[[n^2 + m^2, (n - m)^2, \min(d, d^t)]]$.

14.11.12 Error-correcting conditions, algebraically

Let $\mathcal{S} \leq \mathcal{P}_n$ be a stabiliser group, and let $\mathcal{E} \subseteq \mathcal{P}_n$ be a set of physical errors. Prove that the stabiliser code defined by \mathcal{S} can perfectly correct for all errors in \mathcal{E} if and only if

$$E_1^\dagger E_2 \in N(\mathcal{S}) \setminus \mathcal{S}$$

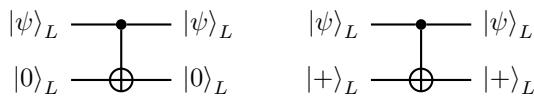
for all $E_1, E_2 \in \mathcal{E}$.

14.11.13 Steane error correction: towards fault tolerance

We have seen how we can measure the stabilisers of a stabiliser code by using a Hadamard test, resulting in ± 1 outcomes. From these, we can determine where the errors are likely to have occurred and then correct them. In Chapter 15, we will see that the scenario of fault-tolerant computation requires us to be very careful with how we construct our error-correcting circuits in order to minimise the propagation of faults that occur during the error-process itself. A particularly useful strategy is to not have multiple qubits interacting, but this contradicts the measurements of the stabilisers, since these, by definition, act on many qubits. There are a variety of ways to deal with this. In the case of CSS codes, Steane himself provided a particularly elegant method, which we will now explore.

Note that, while we are *motivated* by the possibility of there being faults during the error correction, for now we will still assume that the error-correction process proceeds perfectly, and we are only trying to identify and fix errors on the incoming (logical) state $|\psi\rangle_L$.

As we shall later see, CSS codes all have **transversal** c-NOT gates: applying a c-NOT to each physical qubit gives exactly the effect of a c-NOT on the logical qubit. In other words, we can implement the logical operator $c\text{-NOT}_L$ by taking a tensor product of usual controlled-NOT gates. What this means for us right now is that, at the logical level, we can consider circuits such as



1. Verify that the above circuits do indeed have the claimed outputs.

This means that, if the qubits are in the logical space (i.e. have undergone no errors), then the action of the circuit is trivial. So what happens if there's an error? Let's assume that we're working with an $[[n, 1, d]]$ CSS code.

2. If an X or Z error has already affected the input logical state $|\psi\rangle_L$ on a specific physical qubit (say, the i -th) in the circuit on the left above, what are the possible errors on the final state?

We are assuming the availability of the logical states $|0\rangle_L$ and $|+\rangle_L$, but **state preparation** is another challenge that we will eventually have to deal with!

3. If an X or Z error has already affected the input logical state $|\psi\rangle_L$ on a specific physical qubit (say, the i -th) in the circuit on the right above, what are the possible errors on the final state?

In other words, we see that single errors cannot propagate to more than one error on each logical qubit. This will prove to be very useful when thinking about fault tolerance, as the same is also true if errors occur during the circuit. Now we should see how error correction works. Let's consider the circuit on the left above.

4. The X stabilisers are defined by the rows of a parity-check matrix H . We measure each physical qubit of the second logical qubit, $|0\rangle_L$, in the X basis at the end of the circuit. In the absence of any errors, we get a measurement outcome $y \in \{0, 1\}^n$. What is the value of $H \cdot y$?
5. If a weight- w Z -error occurs on the state $|\psi\rangle_L$ before the (transversal) controlled-NOT, where $2w < d$, then what are the possible measurement outcomes? How do we identify which corrections to make?

So this circuit allows us to correct Z -errors on the input, but it does absolutely nothing to X -errors.

6. Show that the circuit on the right above enables the correction of X -errors in a similar way, where the extra logical qubit is now measured by measuring each individual qubit in the Z (i.e. computational) basis.

15 Fault tolerance

TO-DO

This section is not yet finished.

Appendix: Further topics and selected reading

About ...

This section is not yet finished.