Surviving as a Quantum Computer in a Classical World

Daniel Gottesman

January 25, 2024

Contents

Part I

Quantum Error Correcting Codes

Chapter 1

Know Your Enemy: Quantum Errors

In this book, you will learn how to seek out and destroy errors on quantum states. Quantum errors are nasty, unforgiving things. If you don't know what you are doing, a single misstep can result in the destruction of irreplaceable quantum information. Of course, nobody's perfect, and in part II, you will learn how to handle your own fallibility. (Short answer: very carefully.) For now, we will assume you don't make any mistakes, but that doesn't mean things will be easy. The quantum errors are still out there, hungering to consume quantum information. You need to get the errors before they get you.

The key to most error hunts is preparation. There are two forms of preparation. One is dressing your quantum information properly: that is, encoding it in an appropriate quantum error-correcting code. You will learn about that in the other chapters in part I. This chapter will focus on the other aspect of preparation: studying the habits of the errors you are hunting.

1.1 The Quantum Channel

The most common way of characterizing a source of errors in a quantum system is as a quantum channel. "Quantum channel" is a general term for an opportunity for the environment to introduce errors into your quantum system. Usually, we consider a situation as depicted in figure 1.1. Alice wants to send quantum information to Bob. Alice and Bob both have perfect quantum computers, but the connection between them is possibly imperfect: the quantum channel. (We will drop the assumption that Alice and Bob have perfect quantum computers in part II.) While the quantum information is in transit, the external world (the "environment") has a chance to interact with the system, thereby changing it in some way, introducing errors (or "noise") relative to the state that Alice sent.

A common example of a quantum channel is an optical fiber. Single photons can pass through the optical fiber, but they may be lost or altered en route. Other possibilities are sending a photon through the air, physically moving an atom with information encoded in the state of the electron or nuclear spin, successively swapping the states of neighboring qubits arranged on a line, or even using quantum teleportation to send

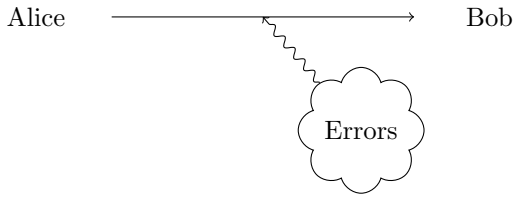

Figure 1.1: Alice, who has a perfect quantum computer, wants to send qubits to Bob, who also has a perfect quantum computer. However, the communications channel between them is *not* perfect.

a quantum state from Alice to Bob with classical communication and some sort of entangled state. This is not an exhaustive list. Indeed, any sort of communication, even a classical telephone call, can be considered as a quantum channel. If you try to send a qubit through a regular telephone connection, no amount of quantum error correction will allow you to recover the full quantum state afterwards, but that doesn't affect the telephone line's status as a quantum channel — it is simply a *very noisy* quantum channel.

Another common situation is when Bob is replaced by Alice's future self. In this case, we really want a "quantum memory": Alice wants to prepare some quantum information, go off and do other things, and then return and manipulate her stored quantum information again. If we assume that Alice's manipulations at the beginning and the end of the process are perfect, we can consider the "memory" portion, when the qubit is stored but subject to noise, as a quantum channel.

It is worth noting that the notion of a quantum channel only applies when we can look at a single communications link in isolation. That is, to have a quantum channel, the quantum state that exits the channel should only depend on the quantum state that goes into the channel. That may seem like a tautology, but it is not. Imagine that Alice has a quantum memory, and prepares a qubit to store in the memory at time 0. At time 1, she returns and fiddles some more with the stored system, then goes away again and comes back at time 2. The storage from time 0 to time 1 is a quantum channel (assuming Alice's initial preparation of the qubit is perfect), but the storage from time 1 to time 2 might not be. The problem is that the environment might remember what happened between time 0 and 1 (and more importantly, might remember something about the *state* that was stored between time 0 and 1) and use that to influence what it does to the state during the second time interval. The error now no longer depends only on what state is stored at time 1; it also depends on what state was stored at time 0. Of course, some environments have a very short memory, and in that case, it is a very good approximation to consider the time interval 1 to 2 to be independent of the time interval 0 to 1, and with that approximation, the second time interval *is* a quantum channel. When the environment has no memory, and every time interval is independent of any other non-overlapping time interval, the environment (or the error source) is called *Markovian*. When the environment does remember over time scales long enough to matter, it is a *non-Markovian* environment.

In part I, we only consider the case depicted in figure 1.1. There the environment has only one opportunity to attack the quantum information. While that opportunity may last for an extended period of time, because Alice and Bob do not do anything with the quantum information during that time, we can lump together everything the environment does to the state into a single transformation, and consider the whole communications line as a single quantum channel. The question of whether the environment is Markovian or non-Markovian then becomes moot. If we were to generalize this picture and allow noise during Alice and Bob's processing of the qubit, then the question arises again, since the environment then gets more than one shot at the quantum information, but don't worry about that situation until part II.

Now it is time to formally define a quantum channel:

Definition 1.1. A *quantum channel* is a completely positive trace-preserving (CPTP) map.

Wasn't that easy? At least, it is if you know what a CPTP map is. If not, you should refer to appendix A, where you will learn that a CPTP map is the most general physically possible transformation for an operation where the output depends only on the input, so this is the right definition. It is frequently convenient to consider the Kraus form of a CPTP map:

$$\mathcal{E}(\rho) = \sum_k A_k \rho A_k^\dagger. \tag{1.1}$$

We can think of this channel as a collection of possible errors A_k, where error A_k occurs with probability $\mathrm{tr}(A_k \rho A_k^\dagger)$. However, note that the probability of A_k occurring is not a single value but actually depends on the state ρ. Furthermore, remember that the decomposition into A_k operators is not unique and that $\sum_k A_k^\dagger A_k = I$.

Frequently, instead of dealing explicitly with a quantum channel, I will instead refer to the *set of possible errors*. One way to write this set is $\mathcal{E} = \{A_k\}$, but it is frequently convenient to rescale the errors, so more generally $\mathcal{E} = \{E_k\}$, where each $A_k = p_k E_k$ for some scalar p_k.

1.2 Single-Qubit Example Channels

1.2.1 Unitary Channels, Pauli Errors

Let us start by discussing some examples of channels acting on a single-qubit input. The simplest case is when there is only a single value of k in the Kraus decomposition:

$$\mathcal{E}(\rho) = A\rho A^\dagger. \tag{1.2}$$

Since $\sum_k A_k^\dagger A_k = I$, it follows that A is unitary. Typographically, I will usually represent a unitary channel the same way as a unitary matrix, e.g. $A(\rho)$ versus $A(|\psi\rangle)$, even though they formally act on different kinds of objects (density matrices versus state vectors). The one exception is that I will usually write the identity channel as $\mathcal{I}$, as opposed to the identity unitary I.

There are of course infinitely many unitary maps, but some are more interesting than others. One set that you will be particularly sick of by the end of this book are the Pauli matrices:

$$I = \begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix}, \quad X = \begin{pmatrix} 0 & 1 \\ 1 & 0 \end{pmatrix}, \quad Y = \begin{pmatrix} 0 & -i \\ i & 0 \end{pmatrix}, \quad Z = \begin{pmatrix} 1 & 0 \\ 0 & -1 \end{pmatrix}. \tag{1.3}$$

I is, of course, the identity matrix — no error. X is probably the first thing you think of when you think of an error, a classical bit flip:

$$X|0\rangle = |1\rangle \tag{1.4}$$

$$X|1\rangle = |0\rangle. \tag{1.5}$$

However, since it is a quantum operator, it also acts sensibly on superpositions:

$$X(\alpha|0\rangle + \beta|1\rangle) = \alpha|1\rangle + \beta|0\rangle. \tag{1.6}$$

Z is the most basic kind of truly quantum error, a phase flip:

$$Z(\alpha|0\rangle + \beta|1\rangle) = \alpha|0\rangle - \beta|1\rangle. \tag{1.7}$$

Y is then just a combined bit flip and phase flip error:

$$Y = iXZ \tag{1.8}$$

$$Y(\alpha|0\rangle + \beta|1\rangle) = i\alpha|1\rangle - i\beta|0\rangle. \tag{1.9}$$

Recall that an *overall phase* — one that affects all states uniformly — has no physical significance, so Y and XZ are really the same channel. In the form presented above, all the Pauli matrices are Hermitian as well as unitary, which is sometimes useful.

In other contexts, the Pauli matrices are often written as σ_x, σ_y, and σ_z or σ_1, σ_2, and σ_3, but those are too much writing and less easy to read. I'll be using the Pauli matrices a *lot* in this book, so I'll use the more straightforward notation X, Y, and Z. In some of the earlier quantum error-correction literature, $Y = XZ$ instead of iXZ. There is not a huge difference, but I think this convention is somewhat nicer overall.

There are many more single qubit unitary errors, and some are even interesting. For instance, we can have phase rotation by an arbitrary angle:

$$R_\theta = \begin{pmatrix} e^{-i\theta} & 0 \\ 0 & e^{i\theta} \end{pmatrix} = e^{-i\theta} \begin{pmatrix} 1 & 0 \\ 0 & e^{2i\theta} \end{pmatrix}. \tag{1.10}$$

Again, we can ignore an overall phase, so R_θ is the same channel as $\mathrm{diag}(1, e^{2i\theta})$. The full set of physically distinct one-qubit unitary channels is the group $\mathsf{SU}(2)$.

In the Bloch sphere picture of the state space for a qubit, a unitary map is just a rotation of the sphere. X, Y, and Z are π rotations around the X, Y, and Z axes, as one might expect. R_θ is a rotation by angle 2θ around the Z axis. (I have defined R_θ this way in order to agree with the prevailing terminology for phase rotations, which results from talking about spin-1/2 particles.) Note that a reflection of the Bloch sphere is not a completely positive map. For instance, the transpose map is a reflection in the XZ plane, and when applied to part of an entangled state, the transpose gives something non-positive.

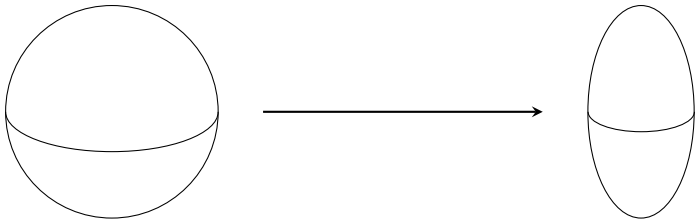

Figure 1.2: Bloch sphere transformation induced by a dephasing channel

1.2.2 Dephasing Channel

One very commonly-encountered channel is the dephasing channel.

Definition 1.2. A channel of the form

$$\mathcal{R}_p(\rho) = (1-p)\rho + pZ\rho Z^\dagger. \tag{1.11}$$

is a *dephasing channel*. When $p = 1/2$, we have the *completely dephasing channel*. We usually restrict attention to $p \leq 1/2$ since channels with $p > 1/2$ are related to channels with $p < 1/2$ by a Z operation.

The Kraus operators of a dephasing channel in this form are $\sqrt{1-p}I$ and $\sqrt{p}Z$. Since both are proportional to unitary maps, the probabilities of these two errors occurring do not depend on the input state. Thus, this channel corresponds to no error with probability $1-p$ and a phase flip with probability p. We can calculate how it acts on the density matrix in component form:

$$\mathcal{R}_p : \begin{pmatrix} a & b \\ c & d \end{pmatrix} \mapsto \begin{pmatrix} a & (1-2p)b \\ (1-2p)c & d \end{pmatrix}. \tag{1.12}$$

In other words, a dephasing channel shrinks the off-diagonal components of the density matrix. In the completely dephasing channel, the off-diagonal terms disappear completely.

An alternate Kraus decomposition is also edifying.

$$\mathcal{R}_p(\rho) = (1-2p)\rho + \frac{2p}{\pi}\int_0^\pi R_\theta \rho R_\theta^\dagger d\theta. \tag{1.13}$$

The dephasing channel is a channel for which, with probability $1-2p$, nothing happens to the state, and with probability $2p$, the phase between $|0\rangle$ and $|1\rangle$ is completely randomized. This seems to conflict with the decomposition into I and Z, where the state is left unchanged with probability $1-p$, not with probability $1-2p$. However, when the dephasing angle θ is chosen uniformly at random, there is a reasonable chance that θ is small and the state does not change very much. Of course, the probability that the angle θ is *exactly* zero is just $1-2p$, but using the magic of quantum mechanics, the small-θ cases cancel out just right so that if we break the channel up into I and Z, we find the probability of error is only p.

This example illustrates a rather disturbing principle: in quantum mechanics, the notion of "probability of error" is inherently somewhat subjective. When error correction is concerned, the decomposition into I and Z is the better choice for the dephasing channel, for reasons that will become clearer in chapter 2. However, there are many channels for which none of the decompositions is particularly favored, even for the specific application of quantum error correction.

In the Bloch sphere picture, a dephasing channel shrinks the sphere into an ellipsoid, leaving the Z axis unchanged, as in figure 1.2. A completely dephasing channel shrinks the Bloch sphere down to just the segment on the Z axis.

The dephasing channel is a physically very interesting channel. A dephasing channel occurs in any system where the environment learns about the qubit in the standard basis but does not otherwise interfere with the state. Even in a more realistic system where there are more complicated interactions between the

system and the environment, there is frequently a large dephasing component to the noise. The prevalence of approximate dephasing channels is one of the reasons that the macroscopic world appears classical to us — a completely dephasing channel converts a qubit into a probabilistic classical bit.

We can make a simple model of a dephasing channel by having one environment qubit interact with the system qubit via a Hamiltonian $H = \omega Z \otimes Z$. The environment qubit starts in the pure state $|+\rangle = (|0\rangle + |1\rangle)/\sqrt{2}$, and the system qubit starts in state $|\psi_0\rangle = \alpha|0\rangle + \beta|1\rangle$. Then, after a time t, the state of the two qubits is

$$e^{-i\omega t Z\otimes Z}|\psi_0\rangle \otimes |+\rangle = \frac{1}{\sqrt{2}}(\alpha e^{-i\omega t}|00\rangle + \alpha e^{+i\omega t}|01\rangle + \beta e^{+i\omega t}|10\rangle + \beta e^{-i\omega t}|11\rangle. \tag{1.14}$$

(Taking $\hbar = 1$.) Tracing over the second (environment) qubit, we find that the system qubit at time t is in an equal mixture of $R_{-\omega t}|\psi_0\rangle$ and $R_{+\omega t}|\psi_0\rangle$, giving it density matrix

$$\begin{pmatrix} |\alpha|^2 & \alpha\beta^* \cos(2\omega t) \\ \alpha^*\beta\cos(2\omega t) & |\beta|^2 \end{pmatrix}. \tag{1.15}$$

In other words, at time t, we have the dephasing channel $\mathcal{R}_{(1-\cos(2\omega t))/2}$. In this simple model, the system dephases at short times, but after a longer time $t = \pi/\omega$, it returns to its starting state.

In a more realistic system, there are many different environment qubits interacting with the system, and/or there are additional qubits interacting with the environment qubits. If we take a Markovian form of our simple model, and assume that the environment's internal interaction effectively resets the environment qubit after a very short time, phase coherence instead decays steadily:

$$\begin{pmatrix} |\alpha|^2 & \alpha\beta^* e^{-t/T_2} \\ \alpha^*\beta e^{-t/T_2} & |\beta|^2 \end{pmatrix}, \tag{1.16}$$

corresponding to a dephasing channel $\mathcal{R}_{(1-e^{-t/t_2})/2}$. The time constant T_2 of the exponential decay is known as the t_2 *time*. (Otherwise I probably would have used a different symbol.) One way to think about this behavior is that there is a constant probability $1/T_2$ per unit time that the phase is completely randomized as an instantaneous "quantum jump".

Another common source of dephasing is a varying energy difference between the $|0\rangle$ and $|1\rangle$ states. If $|a\rangle$ has energy E_a, after time t, $|a\rangle$ has evolved into $e^{-iE_a t}|a\rangle$. We can ignore the global phase, but there is still a relative phase difference $e^{-i(E_1-E_0)t}$ between $|0\rangle$ and $|1\rangle$. However, when E_0 and E_1 are known constants, we can generally ignore the relative phase too: if we keep track of the time t, the relative phase is known, and we can take it into account in any operation we want to perform. In some systems, there is a phase reference (such as the phase of a laser) which automatically compensates for the phase difference. However, when the energy difference varies unpredictably, we can no longer keep precise track of the relative phase, resulting in an uncompensated relative phase shift accumulating randomly over time. This results in dephasing. In some cases, the relative phase shift is not random, but is simply unknown, and more sophisticated techniques may be able to compensate.

Experimentally, the signature of dephasing is often a decay of coherent interference effects. In a Rabi oscillation experiment, the system cycles $\cos(\Omega_R t)|0\rangle + \sin(\Omega_R t)|1\rangle$. After time t, the system is measured to test if it is $|0\rangle$ or $|1\rangle$. If the system were perfect, if we were to plot the probability of 1 against the time, we would get a perfect oscillation $\sin^2(\Omega_R t)$ for all times. Instead, we get something more like figure 1.3, with oscillations decreasing in amplitude.

Let us imagine a Markovian environment, and assume that the T_2 time is much longer than the Rabi frequency Ω_R, so that we can assume the dephasing and oscillation are independent. If there is a quantum jump causing full dephasing at time t_0, the pure state $\cos(\Omega_R t_0)|0\rangle + \sin(\Omega_R t_0)|1\rangle$ becomes the mixed state with probability $\cos^2(\Omega_R t_0)$ of $|0\rangle$ and probability $\sin^2(\Omega_R t_0)$ of $|1\rangle$. Then $|0\rangle$ and $|1\rangle$ continue with their

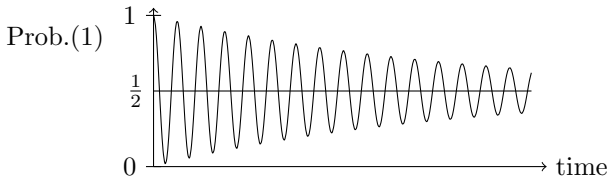

Figure 1.3: Decay of Rabi oscillations due to dephasing

own oscillations, but out of phase with each other. This mixture has density matrix

$$\begin{pmatrix} \cos^2(\Omega_R t_0)\cos^2(\Omega_R t') + \sin^2(\Omega_R t_0)\sin^2(\Omega_R t') & (\cos^2(\Omega_R t_0) - \sin^2(\Omega_R t_0))\sin(\Omega_R t')\cos(\Omega_R t') \\ (\cos^2(\Omega_R t_0) - \sin^2(\Omega_R t_0))\sin(\Omega_R t')\cos(\Omega_R t') & \cos^2(\Omega_R t_0)\sin^2(\Omega_R t') + \sin^2(\Omega_R t_0)\cos^2(\Omega_R t') \end{pmatrix} \tag{1.17}$$

$$= \begin{pmatrix} \sin^2(\Omega_R t_0) + \cos(2\Omega_R t_0)\cos^2(\Omega_R t') & \cos(2\Omega_R t_0)\sin(\Omega_R t')\cos(\Omega_R t') \\ \cos(2\Omega_R t_0)\sin(\Omega_R t')\cos(\Omega_R t') & \sin^2(\Omega_R t_0) + \cos(2\Omega_R t_0)\sin^2(\Omega_R t') \end{pmatrix} \tag{1.18}$$

$$= \sin^2(\Omega_R t_0) I + \cos(2\Omega_R t_0)\rho(t'), \tag{1.19}$$

where $t' = t - t_0$ and $\rho(t')$ is the density matrix of a Rabi oscillation running for time t'. We will still see Rabi oscillation, but with a reduced amplitude.

1.2.3 Depolarizing Channel and Pauli Channel

While the dephasing channel is perhaps the favorite (or least favorite, depending on your point of view) of experimentalists, it lacks a certain quantumness. It only involves one kind of non-trivial error, and indeed, in an appropriate choice of basis, it can be viewed as a purely classical noisy channel. Theorists are instead enamored of the depolarizing channel, which does experience the full range of quantum errors and is highly symmetric. It is not so common in the real world, which is less symmetric than theorists would prefer, but it is still extremely useful for exploring quantum error correction.

Definition 1.3. The *depolarizing channel* is the quantum channel

$$\mathcal{D}_p(\rho) = (1-p)\rho + \frac{p}{3}X\rho X^\dagger + \frac{p}{3}Y\rho Y^\dagger + \frac{p}{3}Z\rho Z^\dagger. \tag{1.20}$$

$\mathcal{D}_{3/4}$ is the *completely depolarizing channel*.

The depolarizing channel has an equal chance of an X, Y, or Z error, each with probability $p/3$. We can analyze what it does to the density matrix:

$$\mathcal{D}_p : \begin{pmatrix} a & b \\ c & d \end{pmatrix} \mapsto \begin{pmatrix} (1-2p/3)a + (2p/3)d & (1-4p/3)b \\ (1-4p/3)c & (1-2p/3)d + (2p/3)a \end{pmatrix}. \tag{1.21}$$

This is not particularly enlightening until you remember that $\operatorname{tr}\rho = 1$, in which case you can see that

$$\mathcal{D}_p(\rho) = (1-4p/3)\rho + (4p/3)(I/2). \tag{1.22}$$

In other words, with probability $1 - 4p/3$, the qubit is left alone, but with probability $4p/3$, it is replaced with the completely mixed state. It also should now be clear why I defined $p = 3/4$ as the completely depolarizing channel: in that case, the input state is always replaced with the maximally mixed state. As with the dephasing channel, there is some ambiguity as to what the "true" error rate is for the depolarizing channel, but the two representations can be reconciled by recognizing that the completely mixed state does contain a component of the original input state.

The other thing to notice about the second representation is that it is much more symmetric than the first one. The decomposition into Paulis has a certain amount of symmetry, treating X, Y, and Z on the

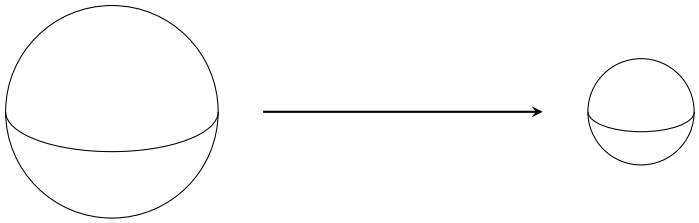

Figure 1.4: Bloch sphere transformation induced by a depolarizing channel

same footing, but the second decomposition has even more: there are no preferred bases or unitary operators at all appearing in it. This is the true beauty of the depolarizing channel — it can at once be considered as a simple mixture of the very basic Pauli errors, but also is invariant under any kind of unitary rotation. This symmetry property suggests a decomposition for the depolarizing channel akin to equation (1.13), and indeed there is one:

$$\mathcal{D}_p(\rho) = (1 - 4p/3)\rho + \int_{\mathsf{SU}(2)} U\rho U^\dagger dU, \tag{1.23}$$

where the integral uses the Haar measure, the unitarily-invariant measure over $\mathsf{SU}(2)$.

We can generalize the depolarizing channel by giving up its symmetry but keeping its decomposition into Paulis:

Definition 1.4. A channel of the form

$$\mathcal{E}(\rho) = p_I\rho + p_X X\rho X^\dagger + p_Y Y\rho Y^\dagger + p_Z Z\rho Z^\dagger \tag{1.24}$$

is a *Pauli channel*.

A Pauli channel has potentially different probabilities for the four Pauli matrices I, X, Y, and Z. They don't *have* to be different — dephasing channels and depolarizing channels are both examples of Pauli channels — but once you've generalized to a Pauli channel, you might as well take advantage of the opportunity to have some variety among the Paulis. Naturally, $p_I + p_X + p_Y + p_Z = 1$ so that the total probability adds up to 1.

The depolarizing channel uniformly shrinks the Bloch sphere into a smaller sphere still centered on the origin, or to a single point (the maximally mixed state) if we have the completely depolarizing channel. A more general Pauli map shrinks the Bloch sphere into an ellipsoid centered on the origin.

1.2.4 Amplitude Damping Channel

Another physically motivated channel is the amplitude damping channel. It occurs, for instance, if $|0\rangle$ and $|1\rangle$ are the ground and excited state of a two-level atom, and the excited state can decay to the ground state by spontaneously emitting a photon. The amplitude damping channel also occurs for a photonic qubit where $|0\rangle$ is no photon and $|1\rangle$ is 1 photon — damping occurs when the photon escapes or is absorbed somewhere along the way. There should be 2 Kraus operators for the amplitude damping channel, one representing the cases when there is no emission (or loss), and one representing the "no-jump" case. It is worthwhile trying to figure out yourself what the form of the Kraus operators should be before looking at the definition below.

Definition 1.5. An *amplitude damping channel* has the following form:

$$\mathcal{A}_p(\rho) = A_0\rho A_0^\dagger + A_1\rho A_1^\dagger, \tag{1.25}$$

where

$$A_0 = \begin{pmatrix} 1 & 0 \\ 0 & \sqrt{1-p} \end{pmatrix}, \quad A_1 = \begin{pmatrix} 0 & \sqrt{p} \\ 0 & 0 \end{pmatrix}. \tag{1.26}$$

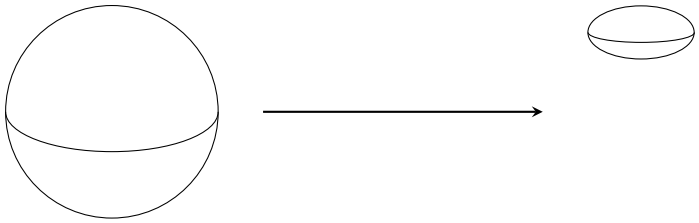

Figure 1.5: Bloch sphere transformation induced by an amplitude damping channel

Probably you were able to get the right form for A_1 (perhaps without the square root, which is a matter of convention depending on how we choose to parametrize amplitude damping channels). It represents the possibility that $|1\rangle$ can become $|0\rangle$. However, $|0\rangle$ never spontaneously gains energy in this idealized channel; even in the real world, it is fairly rare, since it requires that a stray photon of about the right energy be wandering by. Therefore the lower left corner of A_1 is 0.

A_0 might surprise you. (If not, then good work.) The most obvious guess is that since there is no decay, nothing should happen, and A_0 should be proportional to the identity. However, you won't be able to satisfy the constraint that $A_0^\dagger A_0 + A_1^\dagger A_1 = I$ if you choose A_1 as above and $A_0 \propto I$. Conceptually, the reason for this is that A_1 can only occur if the initial state was $|1\rangle$. Therefore, if A_1 *doesn't* happen, it means that the initial state was *more likely* to be $|0\rangle$, and A_0 reflects that, reducing the amplitude of $|1\rangle$ in the initial state. The same phenomenon can be found in classical probability theory, for instance in the "Monty Haul problem."

In the Bloch sphere picture, amplitude damping results in a uniform shrinkage to a smaller sphere. It differs from the depolarizing channel in that the center of the sphere is no longer fixed. Instead, the south pole (the $|0\rangle$ state) is fixed. In the limit $p = 1$, the whole sphere shrinks down to the south pole.

The characteristic decay time for a system undergoing continuous amplitude damping is the "T_1 time." (After all, there had to be a T_1 to go with T_2 for the dephasing time scale.)

1.2.5 Photon Loss Channels

Amplitude damping can occur due to photon loss from a photon channel when the presence or absence of a photon represents $|0\rangle$ or $|1\rangle$. However, this is not a particularly common encoding used for photonic qubits. More often, the qubit is stored in some other state of a single photon (such as the polarization), or by being in one of two modes (a "dual-rail encoding"), or some more complicated structure possibly involving multiple photons. When we use one of these encodings, loss of a photon is no longer represented by an amplitude damping channel.

Consider, for instance, a polarization encoding, with horizontal polarization being $|0\rangle$ and vertical polarization being $|1\rangle$. Now, photon loss can affect either $|0\rangle$ or $|1\rangle$, and what results if the photon does escape is neither of those states, instead being a third state $|\text{vacuum}\rangle$. Therefore, a photon loss channel for polarization encoding takes a qubit input but its output is a 3-dimensional *qutrit*. Assuming both polarizations have equal probability p to lose a photon, the channel is then very simple. There are three Kraus operators, corresponding to no photon loss, loss of a horizontally polarized photon, and loss of a vertically polarized photon:

$$A_0 = 1 - p \begin{pmatrix} 1 & 0 \\ 0 & 1 \\ 0 & 0 \end{pmatrix}, A_1 = p \begin{pmatrix} 0 & 0 \\ 0 & 0 \\ 1 & 0 \end{pmatrix}, \text{ and } A_2 = p \begin{pmatrix} 0 & 0 \\ 0 & 0 \\ 0 & 1 \end{pmatrix}. \tag{1.27}$$

The same channel works for photon loss from a dual-rail encoding.

When the encoding involves multiple photons, there is still a sensible description as a quantum channel, but the analysis requires a bit more quantum optics, and is beyond the scope of this book.

1.2.6 Erasure Errors

The example of the photon loss qubit channel suggests another interesting kind of error. When a photon is not lost, the state is not changed at all. When a photon *is* lost, by monitoring the photon number (but not the polarization), we can, in principle, tell that a photon has been lost. In that case, we do not know what the original state of the system was, but at least we know that something has happened to it. Measuring the photon number without destroying the polarization state is technologically difficult, so for this particular example, this is more an issue of principle than of practice. There are, however, other systems where monitoring for loss of the qubit is easier.

Definition 1.6. A *qubit erasure error* is an error A acting on a single qubit which maps all qubit states to a third state $|\perp\rangle$ orthogonal to the qubit Hilbert space.

Erasure channels may seem to be difficult channels to correct, since the information is completely lost, but that's not the case. By measuring if the state is $|\perp\rangle$ is present (without collapsing superpositions of $|0\rangle$ and $|1\rangle$) — for instance, by measuring the number of photons — we can determine if an erasure error has occurred. Because erasure channels provide some classical information about which qubits underwent errors, erasure channels are actually easier to correct than more general channels.

1.3 Multiple-Qubit Channels

It's hard to do too much with only one qubit. Even if we have only one qubit of data, we'll need more than one qubit to create a real quantum error-correcting code. Therefore, we should also think some about channels acting on multiple qubits.

1.3.1 Pauli Channels

One route to defining multiple-qubit channels is to largely forgot about the tensor product structure and just pick some completely positive map acting on the Hilbert space as a whole. Such channels can be extremely complicated, since they act on a Hilbert space of dimension 2^n when there are n qubits. Sometimes this is necessary, as it reflects the actual physics of the system, but quantum computers are usually built out of systems that naturally break up into qubits, and many-qubit interactions are rare.

Mathematically, it is usually too difficult to deal with an arbitrary n-qubit channel, but the n-qubit generalization of the Pauli channel is sufficiently well-behaved to be sometimes useful.

Definition 1.7. A channel of the form

$$\mathcal{E}(\rho) = \sum_P p_P P \rho P^\dagger, \tag{1.28}$$

where the sum is taken over P which are tensor products of I, X, Y, and Z, is a *Pauli channel*.

In a Pauli channel, the operator P occurs with probability p_P. Pauli channels are a reasonable quantum analogue of classical channels. There is a definite probability of error, and the errors that occur are large and discrete. However, since we can have a mix of bit flip and phase errors, a Pauli channel does have enough quantum features to be interesting.

1.3.2 Independent/Memoryless Channels

The simplest way to create a channel on n qubits is to just treat each qubit separately.

Definition 1.8. An *independent* channel on n qudits (each of dimension q) has the form $\otimes_{i=1}^n \mathcal{E}_i$, where each $\mathcal{E}_i$ is a single-qudit channel.

A dimension-q qudit is a single system whose state space is a Hilbert space of dimension q, so for instance, $q = 2$ gives us a qubit, and for the moment, we will restrict attention to qubits. Often, we set all $\mathcal{E}_i$'s to be equal to $\mathcal{E}$, so that all qubits are treated equally.

As a simple example, we can consider $\mathcal{E} = \mathcal{R}_p$, the dephasing channel. Let us stick to $n = 3$ in order to be more explicit. There are a total of 8 possible Kraus operators for this channel, with the following probabilities:

Probability	Errors
$(1-p)^3$	$I \otimes I \otimes I$
$p(1-p)^2$	$Z \otimes I \otimes I$, $I \otimes Z \otimes I$, $I \otimes I \otimes Z$
$p^2(1-p)$	$Z \otimes Z \otimes I$, $Z \otimes I \otimes Z$, $I \otimes Z \otimes Z$
p^3	$Z \otimes Z \otimes Z$

The total probability of having a Z error on exactly one qubit is then $3p(1-p)^2$, and the probability of having two Z errors is $3p^2(1-p)$.

The chance of having at least one qubit with an error on it is therefore larger than the chance of having a single qubit by itself go wrong under the same dephasing channel $\mathcal{R}_p$. This makes sense, since there are more places for errors to occur. However, when p is small, most of the time, there will only be 0 or 1 Z error, and the other qubits will have I acting on them. This is the case we want to address through quantum error correction — errors are rare, but not negligibly so. In the case of this 3-qubit dephasing channel, we can make a good approximation by considering only the no-error or one-error possibilities, and ignoring the two- and three-qubit error cases.

1.3.3 Definition of t-Qubit Errors

More generally, we can define a t-qubit error to be one that acts on t qubits. However, there are two technical points in exactly how we want to define it:

Definition 1.9. We say that a linear operator A acting on n qubits has *weight* t if it is the tensor product of the identity on $n-t$ qubits with some matrix on the remaining t qubits. The weight of A is denoted wt A. We say that a weight t operator A has *support* on the t qubits on which it acts non-trivially. Generally (but not always), we cite the weight and support for the minimal set of qubits for which A acts non-trivially. A linear operator B acting on n qubits is a *t-qubit error* if it has the form

$$B = \sum_i B_i, \tag{1.29}$$

where each B_i has weight at most t, not necessarily with support on the same set of qubits for different i. An n-qubit quantum channel is a *t-qubit error channel* if it has a Kraus decomposition for which all Kraus operators are t-qubit errors. We can similarly define a *t-qubit error map* acting linearly on n-qubit density matrices but which may not be trace preserving.

Technical point number one is that even though a weight-t operator A acts non-trivially on only t qubits, it can act *arbitrarily* on those t qubits. In particular, it does not need to be a tensor product of errors on the separate qubits — it can entangle them however it likes. For instance, CNOT $\otimes I$ is a weight 2 operator acting on 3 qubits.

Technical point number two is that a t-qubit error B can be the sum of terms acting on different sets of t qubits. This naturally means that more than t qubits will be altered under the action of this error, but it turns out that they are altered in a way that is no more harmful than if we had a channel that had a possibility of altering each set of t qubits as separate Kraus operators. This point will be discussed in greater length in section 2.4.3, once we have a real quantum error-correcting code to examine.

We can also define t-qubit erasure errors. For instance, imagine we have many qubits each stored as the polarization of a photon, and each one undergoes a slight amount of photon loss.

Definition 1.10. A *t-qubit erasure error* is a weight t operator which is the tensor product of erasure errors on the qubits in its support. A *t-qubit erasure channel* is a quantum channel with a Kraus representation

where all Kraus operators are s-qubit erasure errors, with $s \leq t$. s can be different for different Kraus operators.

Note that here I am requiring that each Kraus operator has a specific set of qubits which can be erased, not a superposition of different sets of qubits. This reflects the idea that an erasure is in some sense a classical event, because the set of qubits that were erased can be measured.

1.3.4 Connection Between Independent Channel and t-Qubit Errors

An independent channel seems much more physically plausible (albeit still an approximation) than a t-qubit channel. However, it turns out to be more mathematically straightforward to design quantum error-correcting codes for t-qubit channels (or to correct a set of t-qubit errors) than for independent channels. Luckily, as suggested by the example of the 3-qubit dephasing channel, an n-qubit independent channel can be well approximated by a t-qubit channel (for some $t < n$) when the single-qubit channels making up the n-qubit channel are all close to the identity.

Theorem 1.1. *Let $\mathcal{I}$ be the 1-qubit identity channel and $\mathcal{E} = \otimes_{i=1}^{n} \mathcal{E}_i$ be an n-qubit independent channel, with $\|\mathcal{E}_i - \mathcal{I}\|_\diamond < \epsilon \leq \frac{t+1}{n-t-1}$, and $\epsilon \leq 1/3$ as well. Then*

$$\|\mathcal{E} - \tilde{\mathcal{E}}\|_\diamond < 5\binom{n}{t+1}[(4e+2)\epsilon]^{t+1} \tag{1.30}$$

for some t-qubit error map $\tilde{\mathcal{E}}$.

The significance of the theorem is that if we have a quantum error-correcting code which is designed to correct t-qubit error channels, it will automatically also correct independent channels where the single-qubit tensor factors are sufficiently close to the identity. Actually, that's not completely true: While the theorem does show that, this is not really the significance of the theorem, since there is much easier proof (which we will see in chapter 2) that a quantum error-correcting code that corrects t-qubit errors also corrects small independent error channels. Rather, this theorem provides a motivation for thinking about t-qubit error channels, which might otherwise seem rather bizarre.

When $\|\mathcal{E}_i - \mathcal{I}\|_\diamond < \epsilon$, we are getting about an ϵ chance of error per qubit, so with n qubits, we expect around ϵn errors. Therefore, we would not expect to be able to approximate $\mathcal{E}$ well by a t-qubit error map unless $t \gtrsim \epsilon n$, which is reflected in the bound on ϵ. The other detailed constants in the theorem shouldn't be taken too seriously, as the proof could likely be tightened considerably to get better constants. But if you do insist on taking those constants seriously, you might find, for example, that when $n = 5$ and $t = 1$, you would get $\|\mathcal{E} - \tilde{\mathcal{E}}\|_\diamond \lesssim 8286\epsilon^2$, which only gives a non-trivial bound when ϵ is less than about 0.01.

Two elements of equation (1.30) that *are* significant are the combinatorial factor $\binom{n}{t+1}$, which reflects the number of $(t+1)$-qubit subsets of the n qubits, and the exponent $t+1$ for ϵ, which tells us that the closeness of the approximation improves exponentially as we allow more qubits to have errors. In the limit of large n and any constant ratio t/n, the combination of the combinatorial factor and the exponent means that there will be a threshold value of ϵ below which we get a good approximation for all large n (and indeed, a better one as n gets larger).

Proof. We begin with a lemma on sums of error probabilities or amplitudes that will also be helpful later.

Lemma 1.2. *If $0 < t < n$, then*

a) For any $0 \leq \epsilon \leq 1$, $\sum_{j=t+1}^{n} \binom{n}{j} \epsilon^j (1-\epsilon)^{n-j} \leq \binom{n}{t+1} \epsilon^{t+1}$

b) When $0 \leq \epsilon \leq \frac{t+1}{n-t-1}$, then $\sum_{j=t+1}^{n} \binom{n}{j} \epsilon^j \leq \binom{n}{t+1} (e\epsilon)^{t+1}$

Proof of lemma. There are a number of ways to prove part a. One straightforward method is to interpret ϵ as a probability of some event (which is the main application we will have for this lemma). Then the sum is the probability that the event occurs at least $t+1$ times in n independent trials. We can upper bound

this probability by considering each subset of $t+1$ trials. The total probability of having the event occur in all trials in the subset, without regard to what happens on the other $n-t-1$ trials, is ϵ^{t+1}. There are $\binom{n}{t+1}$ subsets of size $t+1$, so by the union bound, the total probability of having some set of $t+1$ trials with the event is at most $\binom{n}{t+1}\epsilon^{t+1}$. Whenever the event occurs $j > t+1$ times, we have over-counted, since we included that probability as part of all $\binom{j}{t+1}$ sets of size $t+1$ which had the event.

For part b, note that

$$(1-\epsilon)^{n-t-1} \geq \left(1 - \frac{t+1}{n-t-1}\right)^{n-t-1} \geq e^{-(t+1)}. \tag{1.31}$$

Then

$$\sum_{j=t+1}^{n} \binom{n}{j}\epsilon^j = \sum_{j=t+1}^{n} \binom{n}{j}\epsilon^j(1-\epsilon)^{n-j}\frac{1}{(1-\epsilon)^{n-j}} \tag{1.32}$$

$$\leq \sum_{j=t+1}^{n} \binom{n}{j}\epsilon^j(1-\epsilon)^{n-j}\frac{1}{(1-\epsilon)^{n-t-1}} \tag{1.33}$$

$$\leq \binom{n}{t+1}\epsilon^{t+1}e^{t+1} \tag{1.34}$$

by part a and equation (1.31). □

As a warm-up to prove the theorem, let us consider the case when for all i, $\mathcal{E}_i$ has a Kraus operator $(1-\epsilon)I$. In this case, we can say that $\mathcal{E}_i$ has probability ϵ of having an error (one of the other Kraus operators), and a probability of $1-\epsilon$ of having no error. Part a of lemma 1.2 applies, so the probability of having at least $t+1$ errors is at most $\binom{n}{t+1}\epsilon^{t+1}$. In this case, the t-qubit error map $\mathcal{F}$ has all combinations of up to t "error" Kraus operators with the "good" Kraus operator $(1-\epsilon)I$ on the other qubits. The map $\mathcal{F}$ is completely positive, but is not trace preserving, since we have discarded the Kraus operators with more than t errors.

For the general case, first we need a better characterization of single-qubit channels $\mathcal{G}$ which are close to the identity.

Lemma 1.3. *If $\mathcal{E}$ is a quantum channel from $\mathcal{H}_D$ to $\mathcal{H}_D$ satisfying $\|\mathcal{E}-\mathcal{I}_1\|_\diamond < \epsilon \leq 1/3$, then $\mathcal{E}$ has a Kraus representation $\mathcal{E}(\rho) = \sum_k A_k\rho A_k^\dagger$ such that $\|A_0 - I\|_\infty < \sqrt{2D}\,(\epsilon+\epsilon^2) + (\epsilon/2+\epsilon^2)$ and $\sum_{k\neq 0}\|A_k\|_\infty^2 < D\epsilon(1/2+\epsilon)$.*

Proof of lemma. Using the Choi-Jamiolkowski isomorphism, the channel $\mathcal{E}$ corresponds to the entangled state $\Phi_\mathcal{E} = (I\otimes\mathcal{E})(|\Phi^+\rangle\langle\Phi^+|)$, with $|\Phi^+\rangle = \frac{1}{\sqrt{D}}\sum_a |aa\rangle$. (Note that I have normalized the state to make it easier to apply common identities, which is not always the convention used with the Choi-Jamiolkowski isomorphism.) Since $\|\mathcal{E}-\mathcal{I}_1\|_\diamond < \epsilon$,

$$\|\Phi_\mathcal{E} - |\Phi^+\rangle\langle\Phi^+|\|_1 < \epsilon \tag{1.35}$$

as well. Since the trace distance between these two states is small, the fidelity between them is high:

$$\langle\Phi^+|\Phi_\mathcal{E}|\Phi^+\rangle > 1-\epsilon/2. \tag{1.36}$$

Now, $|\Phi^+\rangle\langle\Phi^+|$ has one eigenvalue $+1$ and the remaining eigenvalues 0. Let us also write $\Phi_\mathcal{E}$ in terms of an eigenbasis,

$$\Phi_\mathcal{E} = \sum_{i=0}^{D^2-1}\lambda_i|\phi_i\rangle\langle\phi_i|. \tag{1.37}$$

$\Phi_\mathcal{E}$ is positive and has trace 1, so $\lambda_i \geq 0$ and $\sum_i \lambda_i = 1$. Now consider

$$|\langle\phi_i|\Phi^+\rangle\langle\Phi^+|\phi_j\rangle| = |\langle\phi_i|\left(\Phi_\mathcal{E} - |\Phi^+\rangle\langle\Phi^+|\right)|\phi_j\rangle| < \epsilon \tag{1.38}$$

for $i \neq j$. In addition,

$$\langle \Phi^+ | \Phi_{\mathcal{E}} | \Phi^+ \rangle = \sum_{i=0}^{D^2-1} \lambda_i |\langle \Phi^+ | \phi_i \rangle|^2 > 1 - \epsilon/2. \tag{1.39}$$

We can choose the phase of the eigenstates $|\phi_i\rangle$ to ensure that $\langle \Phi^+ | \phi_i \rangle$ is real and non-negative. Letting $a_i = \langle \Phi^+ | \phi_i \rangle$, we have $a_i \geq 0$, $\sum \lambda_i a_i^2 > 1 - \epsilon/2$, and $a_i a_j < \epsilon$ for $i \neq j$. Assume without loss of generality that a_0 is the largest of the a_is. Then $\sum \lambda_i a_i^2 \leq \sum \lambda_i a_0^2 = a_0^2$, so

$$a_0 > \sqrt{1 - \epsilon/2} \geq 1 - \epsilon/2. \tag{1.40}$$

Thus

$$a_i < \epsilon / a_0 < \frac{\epsilon}{1 - \epsilon/2} \tag{1.41}$$

for $i \neq 0$. Then

$$\sum_{i=0}^{D^2-1} \lambda_i a_i^2 \leq \lambda_0 + \sum_{i=1}^{D^2-1} \lambda_i \frac{\epsilon}{1 - \epsilon/2} \tag{1.42}$$

$$= \lambda_0 + (1 - \lambda_0) \frac{\epsilon}{1 - \epsilon/2} \tag{1.43}$$

$$= \frac{\epsilon + (1 - 3\epsilon/2)\lambda_0}{1 - \epsilon/2}, \tag{1.44}$$

since $\sum \lambda_i = 1$. Therefore,

$$\lambda_0 \geq \frac{(1 - \epsilon/2)^2 - \epsilon}{1 - 3\epsilon/2} \geq 1 - \epsilon/2 - \epsilon^2. \tag{1.45}$$

(assuming $\epsilon \leq 1/3$), which means

$$\sum_{i \neq 0} \lambda_i = 1 - \lambda_0 \leq \epsilon/2 + \epsilon^2 \tag{1.46}$$

for $i \neq 0$.

We have now bounded all the terms we need, but we'd like a tighter bound on a_0. We can do that by going back and plugging in the bound on $\sum \lambda_i$.

$$\| |\phi_0\rangle \langle \phi_0| - |\Phi^+\rangle \langle \Phi^+| \|_1 \leq (1 - \lambda_0) + \| \lambda_0 |\phi_0\rangle \langle \phi_0| - |\Phi^+\rangle \langle \Phi^+| \|_1 \tag{1.47}$$

$$\leq (1 - \lambda_0) + \| \sum_{i \neq 0} \lambda_i |\phi_i\rangle \langle \phi_i| \|_1 + \| \Phi_{\mathcal{E}} - |\Phi^+\rangle \langle \Phi^+| \|_1 \tag{1.48}$$

$$\leq 2\epsilon + 2\epsilon^2. \tag{1.49}$$

But the 1-norm distance between two pure states is just given by

$$\| |\phi_0\rangle \langle \phi_0| - |\Phi^+\rangle \langle \Phi^+| \|_1 = 2\sqrt{1 - |\langle \phi_0 | \Phi^+ \rangle|^2}, \tag{1.50}$$

meaning

$$\sqrt{1 - |\langle \phi_0 | \Phi^+ \rangle|} \leq \frac{\epsilon + \epsilon^2}{\sqrt{1 + |\langle \phi_0 | \Phi^+ \rangle|}} \leq \epsilon + \epsilon^2. \tag{1.51}$$

Thus, in the Choi-Jamiolkowski isomorphism form of the channel, the state has one large eigenvalue, for which the eigenstate is close to the maximally-entangled state, and the other eigenvalues are all small, with the eigenstates far from the maximally entangled state $|\Phi^+\rangle$. (Of course, they could be close to *other* maximally entangled states.) We can recover a set of Kraus operators for the channel by letting

$$A_k |a\rangle = \sqrt{D \lambda_k} \left(\langle a| \otimes I \right) |\phi_k\rangle \tag{1.52}$$

for basis states $|a\rangle$, extended linearly to the full Hilbert space $\mathcal{H}_D$. (Recall that $|\phi_i\rangle$ is a state in $\mathcal{H}_D \otimes \mathcal{H}_D$, so the right-hand side of equation (1.52) is in $\mathcal{H}_D$.) Then $\|A_k\|_1^2 \leq D\lambda_k$, so $\sum_{k\neq 0} \|A_k\|_\infty^2 \leq D\epsilon(1/2+\epsilon)$, as desired.

To get the bound on A_0, note that $A_0|\psi\rangle = \sqrt{D\lambda_0}\langle\psi^*|\phi_0\rangle$, where $|\psi^*\rangle$ is the complex conjugate of $|\psi\rangle$ in the basis $\{|a\rangle\}$. Furthermore, $|\psi\rangle = \sqrt{D}\langle\psi^*|\Phi^+\rangle$. Then

$$\|A_0 - \lambda_0 I\|_\infty = \max_{|\psi\rangle} |A_0|\psi\rangle - \lambda_0|\psi\rangle| \tag{1.53}$$

$$= \max_{|\psi\rangle} \sqrt{D\lambda_0}|\langle\psi^*|\phi_0\rangle - \langle\psi^*|\Phi^+\rangle| \tag{1.54}$$

$$= \sqrt{D\lambda_0}||\phi_0\rangle - |\Phi^+\rangle| \tag{1.55}$$

$$= \sqrt{D\lambda_0}\sqrt{2 - 2\,\mathrm{Re}\langle\Phi_+|\phi_0\rangle} \tag{1.56}$$

$$\leq \sqrt{2D}\,(\epsilon + \epsilon^2), \tag{1.57}$$

applying equation (1.51) in the last line, and recalling we have chosen $\langle\Phi_+|\phi_0\rangle$ to be real. Thus,

$$\|A_0 - I\|_\infty \leq \sqrt{2D}\,(\epsilon + \epsilon^2) + (\epsilon/2 + \epsilon^2) \tag{1.58}$$

□

For the case of qubits and applying $\epsilon \leq 1/3$, the lemma gives $\|A_0 - I\|_\infty \leq 7\epsilon/2$ and $\sum_{k\neq 0} \|A_k\|_1^2 \leq 5\epsilon/3$ for $k \neq 0$. We will round to $\|A_0 - I\|_1 < 4\epsilon$ and $\sum_{k\neq 0} \|A_k\|_1^2 \leq 2\epsilon$ for $k \neq 0$.

Given lemma 1.3, we can use a similar approach for the general case as we did for the warm-up, which was essentially classical. Channel $\mathcal{E}_i$ has Kraus operators A_k^i, with A_0^i close to the identity and A_k^i small for $k \neq 0$. The n-qubit independent channel $\mathcal{E}$ has Kraus operators which are all possible tensor products $\bigotimes_i A_{k_i}^i$. Let $\mathcal{F}$ be the map whose Kraus operators are all tensor products $\bigotimes_i A_{k_i}^i$ with at most t values of i for which $k_i \neq 0$. Then

$$\|\mathcal{F} - \mathcal{E}\|_\diamond \leq \sum_{r=t+1}^{n} \sum_{|S|=r} \prod_{i\in S} \sum_{k_i\neq 0} \|A_{k_i}^i\|_\infty^2. \tag{1.59}$$

The sum over r represents the number of values of i for which $k_i \neq 0$, and S is the set of indices for which $k_i \neq 0$. Since $\sum_{k_i\neq 0} \|A_{k_i}\|_\infty^2 \leq 2\epsilon$, we get

$$\|\mathcal{F} - \mathcal{E}\|_\diamond \leq \sum_{r=t+1}^{n} \binom{n}{r}(2\epsilon)^r \leq \binom{n}{t+1}(2e\epsilon)^{t+1} \tag{1.60}$$

by lemma 1.2.

Now $\mathcal{F}$ is not yet a t-qubit error map because the A_0 terms include some errors. However, the A_0 terms are all near I, so we can expand them as $A_0^i = I + \delta A^i$, with $\|\delta A^i\|_\infty < 4\epsilon$. Given a Kraus operator for $\mathcal{F}$ $A_{\{k_i\}} = \bigotimes_i A_{k_i}^i$ with r indices $k_i \neq 0$, expand all A_0^is in this way and form $A'_{\{k_i\}}$ by discarding all terms in the expansion with more than $t - r$ δA^i factors. The resulting Kraus operators are composed of sums of terms with weight at most t, of which r non-identity factors come from $k_i \neq 0$ terms, and the remaining come from δA^i components of $k_i = 0$ factors.

$$\|A'_{\{k_i\}} - A_{\{k_i\}}\|_\infty \leq \left(\prod_{j|k_j\neq 0} \|A_{k_j}\|_\infty\right) \sum_{s=t+1-r}^{n-r} \sum_{|S|=s} \prod_{i\in S} \|\delta A^i\|_\infty \tag{1.61}$$

$$\leq \left(\prod_{j|k_j\neq 0} \|A_{k_j}\|_\infty\right) \sum_{s=t+1-r}^{n-r} \binom{n-r}{s}(4\epsilon)^s \tag{1.62}$$

$$\leq \binom{n-r}{t+1-r}(4e\epsilon)^{t+1-r} \left(\prod_{j|k_j\neq 0} \|A_{k_j}\|_\infty\right). \tag{1.63}$$

Let ρ be an arbitrary pure state, possibly entangled between $\mathcal{H}_D$ and a reference system. Then

$$\|A'_{\{k_i\}}\rho(A'_{\{k_i\}})^\dagger - A_{\{k_i\}}\rho A^\dagger_{\{k_i\}}\|_1 \leq \|(A'_{\{k_i\}} - A_{\{k_i\}})\rho(A'_{\{k_i\}})^\dagger\|_1 + \|A_{\{k_i\}}\rho((A'_{\{k_i\}})^\dagger - A^\dagger_{\{k_i\}})\|_1 \tag{1.64}$$

$$\leq 2\left(\prod_{j|k_j\neq 0}\|A_{k_j}\|_\infty\right)\|A'_{\{k_i\}} - A_{\{k_i\}}\|_\infty \tag{1.65}$$

$$\leq 2\binom{n-r}{t+1-r}(4e\epsilon)^{t+1-r}\left(\prod_{j|k_j\neq 0}\|A_{k_j}\|^2_\infty\right). \tag{1.66}$$

I have omitted the $\otimes I$ terms affecting the reference system in the first line; the equation is complicated enough as is. In the second line, we have used the property that $\|A|\psi\rangle\| \leq \|A\|_\infty$ and bounded $\|A_{\{k_i\}}\|_\infty$ and $\|A'_{\{k_i\}}\|_\infty$ by the norm of just the terms with $k_j \neq 0$. If we sum over all $\{k_i\}$ with the same locations for which $k_j \neq 0$, we get

$$\sum \|A'_{\{k_i\}}\rho(A'_{\{k_i\}})^\dagger - A_{\{k_i\}}\rho A^\dagger_{\{k_i\}}\|_\infty \leq 2\binom{n-r}{t+1-r}(4e\epsilon)^{t+1-r}(2\epsilon)^r. \tag{1.67}$$

Finally, let $\mathcal{G}$ be the linear map with Kraus operators $A'_{\{k_i\}}$. In $\mathcal{G}$, we have eliminated all Kraus operators with more than t errors. Then we get a bound on $\|\mathcal{G} - \mathcal{F}\|_\diamond$ by applying them to ρ, and considering the 1-norm of the resulting state, which involves summing equation (1.66) over all possible values of $\{k_i\}$ with at most t indices $k_j \neq 0$ (those with more have already been excluded from $\mathcal{F}$). We get

$$\|\mathcal{G} - \mathcal{F}\|_\diamond \leq 2\sum_{r=0}^{t}\binom{n}{r}\binom{n-r}{t+1-r}(4e\epsilon)^{t+1-r}(2\epsilon)^r \tag{1.68}$$

$$= 2\sum_{r=0}^{t}\binom{n}{t+1}\binom{t+1}{r}(4e\epsilon)^{t+1-r}(2\epsilon)^r \tag{1.69}$$

$$\leq 2\binom{n}{t+1}[(4e+2)\epsilon]^{t+1}. \tag{1.70}$$

Combining this with equation (1.60), we get

$$\|\mathcal{G} - \mathcal{E}\|_\diamond \leq 3\binom{n}{t+1}[(4e+2)\epsilon]^{t+1}. \tag{1.71}$$

There is one final step needed. It is possible that $\mathcal{G}$ is no longer completely positive, since it could be that $\|A'_{\{k_i\}}\rho(A'_{\{k_i\}})^\dagger\|_1 > \|A_{\{k_i\}}\rho(A_{\{k_i\}})^\dagger\|_1$, which could result in $\operatorname{tr}\mathcal{G}(\rho) > 1$. We should therefore scale $\mathcal{G}$ down to $C\mathcal{G}$, for appropriately chosen constant C, to get a map that is guaranteed to be completely positive. $\mathcal{F}$ is trace non-increasing, though, so

$$\operatorname{tr}\mathcal{G}(\rho) \leq 1 + \|\mathcal{G} - \mathcal{F}\|_\diamond \leq 1 + 2\binom{n}{t+1}[(4e+2)\epsilon]^{t+1} = 1/C. \tag{1.72}$$

Then

$$\|C\mathcal{G} - \mathcal{E}\|_\diamond \leq 3\binom{n}{t+1}[(4e+2)\epsilon]^{t+1} + 1 - C \tag{1.73}$$

$$\leq 5\binom{n}{t+1}[(4e+2)\epsilon]^{t+1}. \tag{1.74}$$

□

1.4 A Peek Ahead: Errors During Computation

In part II, we'll consider more general types of errors. In particular, quantum gates will be able to go wrong in various ways, and errors will occur multiple times during a computation. As noted above, the quantum channel picture is no longer completely general then, since the noise might be non-Markovian. Mostly we'll stick to Markovian noise, but even then matters are much more complicated. Since our control is no longer reliable, we'll have to deal with errors occurring even while we're trying to fix them. However, there are various subtler difficulties to contend with as well.

For one thing, we don't know when an error occurs, so we can't assume we do error correction immediately after each error. In particular, an error might occur right before a gate that we had intended for some other purpose. Then even if the gate itself works perfectly, it can cause the error to propagate, infecting an additional qubit with the same error. In addition, the effect of the gate can change the type of error. For instance, a Z error that occurs before a Hadamard gate will change into a X error after the gate. This phenomenon makes it much more difficult to take advantage of information we know about the errors. For instance, suppose the noise source is largely dephasing noise. We might want to use a code that is particularly good at correcting dephasing noise, but if we use Hadamard gates in our circuit, some of the Z errors that occur will have become X errors by the time we get around to correcting them, and our code won't work anywhere near as well as expected. A particularly insidious form of this phenomenon occurs when errors happen *during* the implementation of a gate, which, after all, should take a non-zero time. Even if the completed gate does not change the type of error, depending on how the gate is being implemented, the partial gate may in fact alter the structure of the noise.

We'll return to all these issues in part II, and discuss how to design fault-tolerant quantum circuits that allow reliable error correction and computation on encoded states despite the complications.

Chapter 2

Redundancy Without Repetition: Basics Of Quantum Error Correction

Now we are ready to start designing quantum error-correcting codes. A natural place to start for inspiration is to look at the theory of classical error-correcting codes, and indeed it can give us some guidance. However, we'll rapidly see that there are some major differences between classical and quantum error correction.

The simplest classical error-correcting code is the repetition code:

$$0 \mapsto 000 \tag{2.1}$$

$$1 \mapsto 111. \tag{2.2}$$

If we send this 3-bit encoding through a 1-bit error classical channel, it is clear that we will be able to correct for any error that occurs on a single bit. If all three bits are the same, we know there hasn't been an error, while if one of the three is different, for instance 010, we know that the one that's different is the one that's wrong. By enforcing a boring conformity among the bits, we can recover the original state. Recall that if we use an independent channel instead of a 1-bit error channel, then there is some chance of 2 or 3 errors, which would fool us. If there are two errors, the one bit that we think is wrong is actually the only bit that's correct, and our well-meaning attempt to fix it will actually complete the error, making all three bits wrong. Luckily, the chance of two errors occurring is only $O(p^2)$ when the probability of an error on a single bit is p (see section 1.3.2). When p is small, the chance of the encoded state ending up wrong is less than the chance that an unencoded bit can make it unchanged through the channel.

Throughout this chapter, we'll assume we have a t-qubit error channel. This is justified by theorem 1.1, which tells us that if we actually have an independent channel, it is very close to a t-qubit channel, at least when the error rate per qubit is low. We will actually reprove a version of theorem 1.1 with an easier proof and better constants specifically applicable to quantum error-correctiong codes.

To make a quantum error-correcting code, we might want to imitate the classical repetition code, but that instantly runs into a few problems. We'll need to find a somewhat different way to protect our quantum information, one that adds redundancy without repeating the state.

2.1 Quantum Error Correction? Ridiculous!

Why am I so dead-set against a quantum repetition code? Perhaps we could encode

$$|\psi\rangle \mapsto |\psi\rangle|\psi\rangle|\psi\rangle. \tag{2.3}$$

If you've had much quantum information experience, you'll immediately see the problem with this encoding: it is forbidden by the No-Cloning Theorem.

1. No-cloning theorem prohibits repeating a quantum state.
2. Measuring the data while determining the error will destroy superpositions.
3. We must correct Y and Z errors in addition to X errors.
4. We must correct an infinite set of unitaries and also channels which decohere the state.

Table 2.1: Barriers to making a quantum error-correcting code.

Theorem 2.1 (No-Cloning Theorem)**.** *There is no quantum operation which maps an arbitrary state $|\psi\rangle$ to $|\psi\rangle|\psi\rangle$.*

Proof. The problem is linearity. Suppose

$$|0\rangle \mapsto |00\rangle \tag{2.4}$$
$$|1\rangle \mapsto |11\rangle. \tag{2.5}$$

Then, because quantum operations must be linear,

$$|\psi\rangle = \alpha|0\rangle + \beta|1\rangle \mapsto \alpha|00\rangle + \beta|11\rangle. \tag{2.6}$$

However, this is not equal to the cloned $|\psi\rangle$ state:

$$|\psi\rangle|\psi\rangle = \alpha^2|00\rangle + \beta^2|11\rangle + \alpha\beta(|01\rangle + |10\rangle). \tag{2.7}$$

□

Another problem has to do with how we correct errors for the repetition code. Given a 3-bit state coming out of the channel, we'd like to look at it and determine which of the three bits is different from the other two. However, when dealing with quantum states, looks really can kill. Or if not kill, at least decohere, destroying any superposition we have. And without the ability to be in a superposition, a qubit is no better than a classical bit. Somehow, to make a quantum error-correcting code, we'll need some way to correct errors without looking too closely at what we're doing.

Even if we can handle these problems, it's clear that quantum error correction will be more complicated than classical error correction. For a classical channel, basically all that can happen is a bit flip error, but quantum codes are going to need to handle phase flip errors as well, not to mention Y errors. While the 3-qubit repetition code is good at correcting bit flip errors, it doesn't do anything to help correct phase flips. Actually, it makes phase flip errors more common since there are now three qubits which could have phase flip errors instead of only one.

In addition to X, Y, and Z, we will also have to handle an infinite set of unitaries, such as the R_θ errors. Then there are more general channels, such as the dephasing channel or depolarizing channel, which turn pure states into mixed states. How can we come up with a correction operation that will turn the state from a mixed state back into a pure state?

At this point, the prospects for a quantum error-correcting code look bleak. The difficulties we've identified so far are listed in table 2.1. But don't worry — there are solutions to all of these problems. If there weren't, this book would be a lot thinner.

2.2 The 3-Qubit Code(s)

We won't try to handle all of these problems at once. Let's focus first on points 1 and 2. We'll try to make a quantum version of the three-bit repetition code. The quantum code will encode a qubit, but only protect against one kind of error, just like the classical three-bit code.

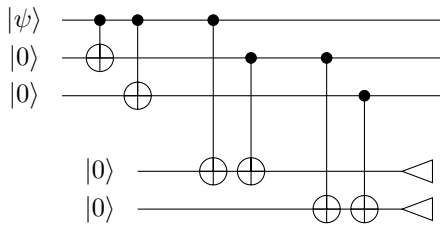

Figure 2.1: Encoding and syndrome measurement circuit for the 3-qubit bit flip correction code.

2.2.1 Correcting Bit Flips for Superposition States

Suppose we apply the classical repetition encoding to the basis states and extend to arbitrary superpositions by linearity:

$$\alpha|0\rangle + \beta|1\rangle \mapsto \alpha|000\rangle + \beta|111\rangle. \tag{2.8}$$

The first thing to notice is that this is an allowed encoding, and doesn't violate the no-cloning theorem. There is no rule against copying basis states — they act just like classical bits. The no-cloning theorem only kicks in if you try to copy superpositions as well. Rather than encoding a superposition $|\psi\rangle$ as three copies of $|\psi\rangle$ (which is impossible), we encode superpositions as *entangled* states. That takes care of difficulty 1.

Let's look at what happens to this state when there's been a bit flip error on the second qubit:

$$X_2(\alpha|000\rangle + \beta|111\rangle) = \alpha|010\rangle + \beta|101\rangle. \tag{2.9}$$

(In this book, I shall use the notation X_2 to indicate that the error X is acting on qubit 2. The same notation applies for gates. However, sometimes I shall omit the subscripts and instead write out a tensor product explicitly; in this case, that would be $I \otimes X \otimes I$.)

As with the classical repetition code, the error can be identified by noticing that the middle qubit is different from the first and third qubits. The critical point is that *this is true for both branches of the superposition.* It is therefore possible to measure the fact that the second qubit is different without measuring whether we have an encoded zero or an encoded one. If we don't measure the data that's encoded in the code, we don't destroy an encoded superposition. This is how we resolve the second barrier: we can measure the error without measuring the information we're trying to preserve.

2.2.2 Encoding Circuit and Error Syndrome Measurement

To understand this point in more detail, look at figure 2.1. The first part of the figure gives the encoding circuit for the 3-qubit code. We start with one qubit in an arbitrary unencoded state and add two additional qubits which become entangled with the first qubit. After the encoding, it is no longer particularly meaningful to ask which was the original data qubit and which are the ones we added. They are now all treated on an equal basis. The encoding of eqnrefeqn:threequbitbitflip is symmetric between all three qubits.

The second part of the circuit contains the error correction process. We wish to know whether one of the qubits is different from the other two, and if so, which one is different. We can break that down into two pieces: We ask whether the first two qubits are the same or different, and then we ask whether the second and third qubits are the same or different. In other words, we wish to know the *parity* of the first two qubits and the parity of the last two qubits. We will store the answers to these two questions on two more extra qubits. Extra qubits like these that are used for this or any other helpful purpose during a computation are known as *ancilla* qubits.

To tell whether two qubits are the same or different, the circuit in figure 2.1 uses two CNOT gates. If they are the same, either neither CNOT flips the corresponding ancilla, or both do; if they are different, exactly one of the CNOT gates flips the ancilla qubit. Thus, when we measure the ancilla qubit for one of the parity measurements, the output we get is equal to the parity: 0 for same (even parity), and 1 for

opposite (odd parity). Notice that the result does not depend at all on the encoded state, simply whether there is a bit flip error on the qubits we are measuring. That means the measurement can be done without disturbing a superposition of the encoded data.

Together, the measurement results for the two ancillas form a bit string known as the *error syndrome*. The error syndrome encapsulates all the information we have about the error. For instance, when the bit flip error is X_2, the error syndrome is 11 because both the first pair and the second pair of qubits are different. Similarly, error syndromes 10 and 01 correspond to the errors X_1 and X_3, respectively, and error syndrome 00 tells us that there is no error. Thus, every possible error for a one-qubit bit flip channel is accounted for. Once we know what the error is, we can simply correct it by performing another bit flip on the appropriate qubit.

Notice that the choice of error correction circuit and corresponding error syndrome is not unique. For instance, we could have measured the parity of the first and third qubits instead of measuring the parity of the second and third qubits. This would have given us the same information, but the correspondence between error syndromes and errors would have been shuffled. We could have measured the parities of all three pairs of qubits, but in that case the error syndrome (which would be 3 bits long) would be redundant: from any two error syndrome bits, we could deduce the third. It is not a coincidence that the number of syndrome bits we need is equal to the number of extra qubits we added to the data in order to perform the original encoding. This is a general property of quantum error-correcting codes, as discussed in chapter 3, although there are certain cases where we don't need all of the information encoded in the syndrome. (See section 17.2 for a description of that case.)

2.2.3 Phase Error Correction

The three-qubit code described above corrects a single bit flip error, but cannot correct any phase flip errors. It is also straightforward to make a code that works the other way: it corrects a single phase flip error, but cannot correct any bit flip errors.

The key to making this code is to notice that the Hadamard transform H switches the role of X and Z errors:

$$|0\rangle \leftrightarrow |+\rangle = \frac{1}{\sqrt{2}}(|0\rangle + |1\rangle) \tag{2.10}$$

$$|1\rangle \leftrightarrow |-\rangle = \frac{1}{\sqrt{2}}(|0\rangle - |1\rangle) \tag{2.11}$$

X acting on the $|0\rangle$, $|1\rangle$ basis is a bit flip, but in the $|+\rangle$, $|-\rangle$ basis, it maps

$$X|+\rangle = |+\rangle \tag{2.12}$$

$$X|-\rangle = -|-\rangle, \tag{2.13}$$

acting as a phase flip. Similarly, Z switches $|+\rangle$ and $|-\rangle$, so it acts as a bit flip in the Hadamard-rotated basis.

Therefore, we can make a three-qubit *phase* correcting code by just applying H to every qubit in the three-qubit bit flip correcting code:

$$|0\rangle \mapsto |\overline{0}\rangle = |+\rangle|+\rangle|+\rangle \tag{2.14}$$

$$|1\rangle \mapsto |\overline{1}\rangle = |-\rangle|-\rangle|-\rangle, \tag{2.15}$$

extended by linearity so that $\alpha|0\rangle + \beta|1\rangle \mapsto \alpha|\overline{0}\rangle + \beta|\overline{1}\rangle$. In this book, I shall use lines over a state or operator to indicate the encoded version of the object.

If there is a single phase flip error, for instance, Z_2, one of the three qubits will be different than the other two in the Hadamard basis. E.g., $Z_2|\overline{0}\rangle = |+\rangle|-\rangle|+\rangle$. We can locate the error in just the same way as before, except that now we should rotate the control qubits for the CNOTs in the error correction circuit by a Hadamard transform to work in the correct basis. The modified encoding and error correction circuit is pictured in figure 2.2.

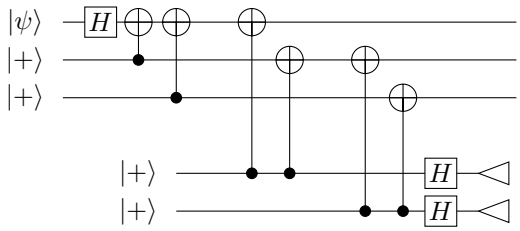

Figure 2.2: Encoding and syndrome measurement circuit for the 3-qubit phase correction code.

2.3 The 9-Qubit Code

The three-qubit codes resolve the first two barriers from table 2.1, but to address the third difficulty, we'll have to add more qubits. In order to make a code that corrects a bit flip error *or* a phase error, we'll encode one qubit into nine qubits, mixing together the encodings from both the bit and phase correcting three-qubit codes:

$$|0\rangle \mapsto |\overline{0}\rangle = \frac{1}{2\sqrt{2}}(|000\rangle + |111\rangle)(|000\rangle + |111\rangle)(|000\rangle + |111\rangle) \tag{2.16}$$

$$|1\rangle \mapsto |\overline{1}\rangle = \frac{1}{2\sqrt{2}}(|000\rangle - |111\rangle)(|000\rangle - |111\rangle)(|000\rangle - |111\rangle). \tag{2.17}$$

Again, we extend the encoding by linearity to work on superpositions. The 9-qubit code was discovered by Peter Shor and so is sometimes called the *Shor code*.

For the nine-qubit code, we will let the set of possible errors be $\{I, X_i, Y_i, Z_i | i = 1, \ldots, 9\}$, so we can have any single-qubit Pauli error. If we have an X error acting on a single qubit, that can be detected by looking at a single group of three qubits to see if one qubit is different from the other two in the standard basis. For instance,

$$X_5|\overline{0}\rangle = \frac{1}{2\sqrt{2}}(|000\rangle + |111\rangle)(|010\rangle + |101\rangle)(|000\rangle + |111\rangle), \tag{2.18}$$

and the error correction circuit of figure 2.1 applied to the middle set of three qubits will identify the error correctly. As before, once we identify the location of an X error, we can correct it by simply flipping the appropriate bit back to its original state. Also as before, the circuit tells us nothing about the encoded state.

If we have a Z error on a single qubit, matters are a little more complicated, but work basically the same way.

$$Z_5|\overline{0}\rangle = \frac{1}{2\sqrt{2}}(|000\rangle + |111\rangle)(|000\rangle - |111\rangle)(|000\rangle + |111\rangle). \tag{2.19}$$

Now we need to compare the three phases. Simply rotating into the Hadamard basis will not quite do it this time (although it gets us close), since we have to take into account the fact that the phase is distributed among a set of three qubits.

We will return to the nine-qubit code in chapter 3, and discuss exactly what to measure to find the error syndrome, but for now, let us just notice that it is possible in principle to make the desired measurement. The set of all correctly encoded states (the *code space*) forms a 2-dimensional subspace of the 2^9-dimensional Hilbert space of nine qubits. The two-dimensional subspace where the first phase is different from the other two (i.e., we have $-++$ or $+--$) is spanned by $Z_1|\overline{0}\rangle$ and $Z_1|\overline{1}\rangle$, and is orthogonal to the code space. Similarly, the subspaces where the second phase is different or the third phase is different are also orthogonal to the code space. Furthermore, these three erroneous code spaces are all orthogonal to each other. Thus, there is a measurement we can make that will distinguish them, and identify whether we have no phase error or one phase error, and if there is a phase error, on which set of three qubits it has occurred. Once we know the correct set of three, we can undo the phase error by applying Z to one of the qubits in that set of three.

A little consideration will show you that this argument applies to Y errors as well. The X and Z error correction procedures are essentially independent, and the argument that we can identify the block of three qubits with a Z error applies equally well if there is an X error on one of the nine qubits. Thus, the nine-qubit code can correct for both an X and a Z error. In particular, it can correct for a single-qubit Y error, which corresponds to having an X and a Z on the same qubit. In the set of possible errors, we only allowed single-qubit errors, but actually the code will still work if you extend the set of possible errors to allow one X error and one Z error on *any* pair of qubits.

2.4 Correcting General Errors

The nine-qubit code addresses problem 3, but it suggests that problem 4 might be troublesome. In order to correct a single bit-flip error, we needed 3 qubits, but to correct Y and Z errors as well, we went up to nine qubits. To solve problem 4 and correct general single-qubit errors, we'll need to handle an infinite set of errors. Does this mean we will have to use infinitely many qubits? Hopefully not.

2.4.1 Continuous Phase Rotation Example

Let's examine the specific case of the continuous phase rotation R_θ in more detail. We can rewrite

$$R_\theta = \begin{pmatrix} e^{-i\theta} & 0 \\ 0 & e^{i\theta} \end{pmatrix} = \cos\theta I - i\sin\theta Z. \tag{2.20}$$

Now suppose the usual nine-qubit code has experienced an error $(R_\theta)_i$ instead of the usual X, Y, or Z error, but imagine that we do not know that and use the usual error correction circuit for the nine-qubit code. We can figure out what happens to state by again applying the linearity of quantum mechanics.

$$(R_\theta)_i|\overline{\psi}\rangle = \cos\theta I|\overline{\psi}\rangle - i\sin\theta Z_i|\overline{\psi}\rangle. \tag{2.21}$$

We know what happens to $I|\overline{\psi}\rangle$ and $Z_i|\overline{\psi}\rangle$ under the error correction circuit: First, we interact with some ancilla qubits and determine the error syndrome corresponding to these errors, then we measure the ancilla qubits. If $|I\rangle_{\text{syn}}$ and $|Z_i\rangle_{\text{syn}}$ are the error syndromes corresponding to the errors I and Z_i, when we interact with the ancillas, we get the following:

$$(R_\theta)_i|\overline{\psi}\rangle \mapsto \cos\theta I|\overline{\psi}\rangle|I\rangle_{\text{syn}} - i\sin\theta Z_i|\overline{\psi}\rangle|Z_i\rangle_{\text{syn}}. \tag{2.22}$$

Right now, we have an entangled state between the code qubits and the ancilla qubits. When we measure the ancilla register, we'll collapse the superposition, getting one of two results:

$$\text{Prob.}(\cos^2\theta): \ I|\overline{\psi}\rangle|I\rangle_{\text{syn}} \tag{2.23}$$

$$\text{Prob.}(\sin^2\theta): \ Z_i|\overline{\psi}\rangle|Z_i\rangle_{\text{syn}}. \tag{2.24}$$

Something miraculous has happened: now we have either an I error (i.e., no error at all) or a Z error, and the outcome of the error syndrome tells us which and where the error is. We can then correct it in the usual way. Somehow, by pretending that we had an I, X, Y, or Z error, we made it actually true that the error was one of those. It's a triumph of wishful thinking.

For the nine-qubit code, the same procedure works with arbitrary single-qubit unitaries, and actually works for any Kraus operator A_k. Indeed, this is not a property of just the nine-qubit code, but of quantum error-correcting codes in general: correcting I, X, Y, and Z errors is sufficient to correct general errors. At this point, we could easily prove this for the nine-qubit code, but it is worthwhile to prove the generalization. To do that, we will want to first define precisely what a quantum error-correcting code is.

2.4.2 Definition of a Quantum Error Correcting Code

Generally speaking, a quantum error-correcting code is a subspace of a larger Hilbert space. Typically, that subspace has been chosen in some complicated entangled way in order to have special properties when errors occur on states from the code space. Therefore, it is most sensible to define a quantum error-correcting code together with the set of errors it corrects. Also, we'll frequently want to consider the subspace as a Hilbert space encoding some quantum data, so we'll define the subspace as a map from a smaller Hilbert space into the big Hilbert space.

Definition 2.1. A *quantum error-correcting code* $(U, \mathcal{E})$ (or *QECC* for short) is a partial isometry $U : \mathcal{H}_K \to \mathcal{H}_N$ with the set of *correctable errors* $\mathcal{E}$ (consisting of linear maps $E : \mathcal{H}_N \to \mathcal{H}_M$) with the following property: $\exists$ quantum operation $\mathcal{D} : \mathcal{H}_M \to \mathcal{H}_K$ such that $\forall E \in \mathcal{E}$, $\forall |\psi\rangle \in \mathcal{H}_K$,

$$\mathcal{D}(EU|\psi\rangle \langle\psi|U^\dagger E^\dagger) = c(E, |\psi\rangle)|\psi\rangle \langle\psi|. \tag{2.25}$$

U is known as the *encoding* operation for the code (or the *encoder*), and $\mathcal{D}$ is the *decoding* map (or *decoder*). $\mathcal{D}$ is also known as the *recovery* operation. We say that the QECC *corrects* E if $E \in \mathcal{E}$.

Sometimes, we don't consider a specific encoding map, and refer to Image(U) (more precisely known as the *code space*) as the QECC. A *codeword* is any state in the code space. Often, a QECC is mentioned without an explicit set of correctable errors, which should be determined by context. Sometimes the error set simply does not matter. When it does matter but is not specified, the set of correctable errors is often the set of all t-qubit errors for the largest possible t; except sometimes it is instead the set of the most likely errors in the system. $\mathcal{H}_K$ is sometimes referred to as the *logical* Hilbert space and $\mathcal{H}_N$ is the *physical* Hilbert space. If $\mathcal{E}$ is the set of all t-qubit errors, we say that U (or Image(U)) is a *t-error correcting code*, or that it *corrects t errors*.

In the above definition (and elsewhere in the book), $\mathcal{H}_D$ is a Hilbert space of dimension D. A partial isometry is like a unitary in that it preserves inner products, but might not be invertible because it may map a Hilbert space to a Hilbert space of strictly larger dimension. Of course $\mathcal{D}$ actually maps matrices on $\mathcal{H}_M$ to matrices on $\mathcal{H}_K$. Since E can be a general linear map, not necessarily unitary, $EU|\psi\rangle$ might not be normalized properly, which is why we need the $c(E, |\psi\rangle)$ factor in equation (2.25). The definition explicitly allows $c(E, |\psi\rangle)$ to depend on $|\psi\rangle$, but it turn outs that it does not (see section 2.5). It does depend on E, however.

To prove this, and in many other contexts, it is helpful to treat the decoding map as a unitary. This can be done via the Stinespring dilation, by purifying $\mathcal{D}$. To do so, we must add an ancilla register. Let the input ancilla to $\mathcal{D}$ be of dimension D and let the output ancilla have dimension D'. Then $\mathcal{H}_M \otimes \mathcal{H}_D \cong \mathcal{H}_K \otimes \mathcal{H}_{D'}$. $\mathcal{H}_{D'}$ plays the role of the error syndrome. Call the purification $V : \mathcal{H}_M \otimes \mathcal{H}_D \to \mathcal{H}_K \otimes \mathcal{H}_{D'}$. Now, $U|\psi\rangle = |\overline{\psi}\rangle$, and when the QECC corrects $E \in \mathcal{E}$,

$$V(E|\overline{\psi}\rangle_N \otimes |0\rangle_D) = \sqrt{c(E, |\psi\rangle)}\, |\psi\rangle_K \otimes |A(E, |\psi\rangle)\rangle_{D'} \tag{2.26}$$

I have put subscripts on the kets to help you keep track of which Hilbert spaces they belong to. There might be a phase on the RHS, but it can be absorbed into the ancilla state $|A(E, |\psi\rangle)\rangle$.

Proposition 2.2. *In definition 2.1, $c(E, |\psi\rangle)$ is independent of $|\psi\rangle$, as is $|A(E, |\psi\rangle)\rangle$ when we purify the decoder.*

Proof. Consider two codewords $|\overline{\psi}\rangle$ and $|\overline{\phi}\rangle$. Imagine we purify the decoder to the unitary V, so

$$VE|\overline{\psi}\rangle \otimes |0\rangle = \sqrt{c(E, |\psi\rangle)}\, |\psi\rangle \otimes |A(E, |\psi\rangle)\rangle \tag{2.27}$$

$$VE|\overline{\phi}\rangle \otimes |0\rangle = \sqrt{c(E, |\phi\rangle)}\, |\phi\rangle \otimes |A(E, |\phi\rangle)\rangle. \tag{2.28}$$

By linearity and the definition of a QECC applied to the superposition codeword $\alpha|\overline{\psi}\rangle + \beta|\overline{\phi}\rangle$,

$$VE(\alpha|\overline{\psi}\rangle + \beta|\overline{\phi}\rangle) \otimes |0\rangle = \alpha\sqrt{c(E, |\psi\rangle)}\, |\psi\rangle \otimes |A(E, |\psi\rangle)\rangle + \beta\sqrt{c(E, |\phi\rangle)}\, |\phi\rangle \otimes |A(E, |\phi\rangle)\rangle \tag{2.29}$$

$$= \sqrt{c(E, \alpha|\psi\rangle + \beta|\phi\rangle)}\, (\alpha|\psi\rangle + \beta|\phi\rangle) \otimes |A(E, \alpha|\psi\rangle + \beta|\phi\rangle)\rangle. \tag{2.30}$$

These two expressions can only be equal if $|A(E,|\psi\rangle)\rangle = |A(E,|\phi\rangle)\rangle = |A(E)\rangle$ (or the discarded ancilla will be entangled with the output state) and $c(E,|\psi\rangle) = c(E,|\phi\rangle) = c(E)$ (or the decoded state for the superposition will be wrong). The conceptual point here is that in order to preserve the coherent superposition in the decoder's output, there cannot be any information about the encoded state left in the ancilla, and the amplitudes (and thus norms) of different encoded states have to match, or the Hilbert space will get distorted. □

Note that the set of correctable errors is not a unique invariant property of an encoding map or code space, which is why I included it as part of the definition. The same code space could be used to correct different sets of errors. For instance, the 3-qubit phase correction code can correct the set $\mathcal{E} = \{I, Z_1, Z_2, Z_3\}$, but it also corrects the set $\mathcal{E} = \{I, Z_1Z_2, Z_1Z_3, Z_2Z_3\}$. In the former case, we interpret the non-zero error syndromes as caused by a single phase error, whereas in the latter case, we only consider two-qubit errors. Since Z_1 has the same error syndrome as Z_2Z_3 but they correspond to different logical states, we have to make a choice between them and cannot include both in the set of correctable errors at the same time. That is, $\mathcal{E} = \{I, Z_1, Z_2Z_3\}$ is *not* a possible set of correctable errors for the 3-qubit phase correction code.

There are many variations of the terms defined above. For instance, code space is sometimes coding space, code subspace, etc., and sometimes it is the encoded subspace. However, the encoded subspace also sometimes refers to $\mathcal{H}_K$, which can also be called the data or encoded data. As defined above, $\mathcal{D}$ incorporates both the error correction procedure and the "unencoding" process of returning the data to $\mathcal{H}_K$, but sometimes they are considered separately.

The decoder $\mathcal{D}$ incorporates whatever processing is necessary to correct and decode the state. For instance, it may incorporate a measurement of the error syndrome, and application of a correction operation conditional on the classical bits resulting from the syndrome measurement. The decoder $\mathcal{D}$ may not, however, be unique. Its behavior is completely determined on the subspace spanned by $E|\overline{\psi}\rangle$, where $E \in \mathcal{E}$ and $|\overline{\psi}\rangle$ is a codeword, but outside of that subspace, $\mathcal{D}$ can act arbitrarily, and the distinctions between different decoders can be important in a number of contexts, such as when studying the efficiency of the decoder, or its behavior on errors outside $\mathcal{E}$, or when considering fault tolerance.

Note that if we define a QECC just as a subspace instead of an encoding map, we haven't lost much information. In particular, the set of correctable errors is the same for all encoders:

Proposition 2.3. *Suppose we have two different encoders $U_1, U_2 : \mathcal{H}_K \to \mathcal{H}_N$ with* $\mathrm{Image}(U_1) = \mathrm{Image}(U_2)$. *Then $(U_1, \mathcal{E})$ is a QECC iff $(U_2, \mathcal{E})$ is a QECC.*

Proof. Because U_1 and U_2 are partial isometries with the same image, they can only differ by a unitary $V : \mathcal{H}_K \to \mathcal{H}_K$:

$$U_2 = U_1 V. \tag{2.31}$$

If we use decoder $\mathcal{D}_1$ for encoder U_1, then $V^\dagger \circ \mathcal{D}_1$ will serve as the decoding map for U_2:

$$\begin{aligned} V^\dagger \mathcal{D}_1(EU_2|\psi\rangle\langle\psi|U_2^\dagger E^\dagger)V &= V^\dagger \mathcal{D}_1(EU_1V|\psi\rangle\langle\psi|V^\dagger U_1^\dagger E^\dagger)V & (2.32)\\ &= V^\dagger(c(E,V|\psi\rangle)V|\psi\rangle\langle\psi|V^\dagger)V & (2.33)\\ &= c(E,|\psi\rangle)|\psi\rangle\langle\psi|. & (2.34)\end{aligned}$$

□

In the definition of a QECC, we have let the dimensions K, M, and N of the various Hilbert spaces involved be arbitrary, although $N \geq K$ or no QECC is possible. Nevertheless, there are some common restrictions. Frequently we assume $K = 2^k$ and $N = 2^n$, so there are k *logical qubits* (or *encoded qubits*) and n *physical qubits*. Sometimes we work with q-dimensional qudits instead of qubits, so $K = q^k$ and $N = q^n$. Occasionally we have even more general situations, where K and N might use different bases or not be powers at all. When there is a tensor factorization of the overall Hilbert space, but I don't want to be too specific about whether the individual factors are qubits, qudits, or maybe even different sizes from each other, I will refer to the *registers*, with each register being one tensor factor. A single error then affects a single register, whatever its size.

Most often $M = N$, so errors map the physical Hilbert space to itself. There is little lost by assuming this, since we can generally ignore any unused states in $\mathcal{H}_N$ or $\mathcal{H}_M$ without negative consequence. The main time when it matters is when errors occur repeatedly on the state before we perform error correction, as happens, for instance, in fault-tolerant quantum computation. In that case, you really need $M = N$, so that you can sensibly apply the same error over and over again.

One special case worth mentioning is that of erasure errors. Recall that an erasure error formally maps a qubit into a qutrit, so to treat erasure errors in the most precise way, we would take $N = 2^n$ and $M = 3^n$. However, most often we imagine that a "stop-leak" gate of some sort is performed before the decoder. The stop-leak gate will map the third $|\perp\rangle$ state of each qutrit to some state, perhaps a random one, of the corresponding qubit. Then we can consider an erasure error as mapping a qubit to a qubit, but with some additional classical information indicating that the qubit has undergone an error.

This is maybe a good place for an extremely important digression on the terminology of error correction. Specifically, I want to discuss the use of hyphens. Wait, did I say "important?" I meant "unimportant." But I am going to discuss hyphens anyway. In English, hyphens are occasionally used in compound nouns but not that frequently. Where they *are* used is to make compound adjectives. Thus, we have a hyphen in "error-correcting code" and "fault-tolerant computation" but not in "error correction" and "fault tolerance," unless one of the component words happens to get split between lines. (Notably, though, there is no hyphen for the "quantum" part, so "quantum error-correcting code.") Certainly, many scientists are not native speakers of English, so they have a good excuse for making this mistake. But you no longer have an excuse.

2.4.3 Linearity of Quantum Error Correction

Now we are ready to generalize the phenomenon we observed in section 2.4.1.

Theorem 2.4. *If $(U, \mathcal{E})$ is a QECC, then $(U, \mathcal{E}')$ is a QECC, where $\mathcal{E}' = \operatorname{span} \mathcal{E}$ is the linear span of $\mathcal{E}$.*

In other words, if a QECC can correct E and F, then it can also correct $\alpha E + \beta F$.

Proof. Basically, the idea is to duplicate the argument used in section 2.4.1 for the general case. We haven't defined the error syndrome for a general QECC, but we do have the decoding map to work with. We will purify it to V so that we can work with only unitary maps.

The QECC corrects $E, F \in \mathcal{E}$, so

$$V(E|\overline{\psi}\rangle_M \otimes |0\rangle_D) = c_E|\psi\rangle_K \otimes |s_E\rangle_{D'} \tag{2.35}$$

$$V(F|\overline{\psi}\rangle_M \otimes |0\rangle_D) = c_F|\psi\rangle_K \otimes |s_F\rangle_{D'}. \tag{2.36}$$

One difference from section 2.4.1 is that $|s_E\rangle$ and $|s_F\rangle$ don't need to be orthogonal. Also note that the c's from definition 2.1 would be the squares of c_E and c_F.

By linearity,

$$V[(\alpha E + \beta F)|\overline{\psi}\rangle_M \otimes |0\rangle_D] = \alpha c_E|\psi\rangle_K \otimes |s_E\rangle_{D'} + \beta c_F|\psi\rangle_K \otimes |s_F\rangle_{D'} \tag{2.37}$$

$$= |\psi\rangle_K \otimes (\alpha c_E|s_E\rangle_{D'} + \beta c_F|s_F\rangle_{D'}). \tag{2.38}$$

Tracing out $\mathcal{H}_{D'}$, we find that the decoder map is of the desired form, with

$$c(\alpha E + \beta F, |\psi\rangle) = \|\alpha c_E|s_E\rangle + \beta c_F|s_F\rangle\|^2. \tag{2.39}$$

□

2.4.4 Correcting Paulis Implies Correcting All Errors

As a corollary of theorem 2.4, we can simplify the condition for a code to correct t errors:

Corollary 2.5. *If a QECC set of correctable errors includes all tensor products of I, X, Y, and Z of weight t or less, it is a t-error correcting code.*

Proof. The set of all errors with support on a given set of t qubits is the set of $2^t \times 2^t$ matrices acting on those qubits. Tensor products of I, X, Y, and Z acting on those t qubits form a basis for this set of matrices. Therefore, an arbitrary weight t error can be written as the sum (with appropriate coefficients) of weight t tensor products of I, X, Y, and Z, and an arbitrary t-qubit error can be written as a sum weight t errors. The corollary then follows from theorem 2.4. □

As a consequence, we find that the nine-qubit code is a 1-error correcting code. Applying the definitions, we find that it encodes one logical qubit into nine physical qubits.

2.4.5 QECCs and Error Channels

We have finally answered all four objections to the existence of quantum error-correcting codes. However, it is worth thinking a little more carefully about what happens when we have a decoherent error that maps pure states to mixed states. For example, suppose that we have a dephasing channel with error probability p acting on qubit i. Then the pure codeword state $|\overline{\psi}\rangle$ becomes:

$$|\overline{\psi}\rangle\langle\overline{\psi}| \mapsto (1-p)I|\overline{\psi}\rangle\langle\overline{\psi}|I + pZ_i|\overline{\psi}\rangle\langle\overline{\psi}|Z_i, \tag{2.40}$$

which is a mixture of the pure states $|\overline{\psi}\rangle$ and $Z_i|\overline{\psi}\rangle$. Now, we know that the full error correction procedure corrects these two pure states:

$$|\overline{\psi}\rangle \mapsto |\overline{\psi}\rangle|I\rangle_{\text{syn}} \tag{2.41}$$

$$Z_i|\overline{\psi}\rangle \mapsto |\overline{\psi}\rangle|Z_i\rangle_{\text{syn}} \tag{2.42}$$

By the linearity of density matrices, that means that the final state after the dephasing channel followed by error correction is:

$$(1-p)|\overline{\psi}\rangle\langle\overline{\psi}| \otimes |I\rangle\langle I|_{\text{syn}} + p|\overline{\psi}\rangle\langle\overline{\psi}| \otimes |Z_i\rangle\langle Z_i|_{\text{syn}} = |\overline{\psi}\rangle\langle\overline{\psi}| \otimes \left[(1-p)|I\rangle\langle I|_{\text{syn}} + p|Z_i\rangle\langle Z_i|_{\text{syn}}\right]. \tag{2.43}$$

That is, the decoded logical qubit ends up as a pure state, but the overall state is still mixed: the randomness introduced by the error ends up in the error syndrome, or in the ancilla Hilbert space $\mathcal{H}_{D'}$ in the proof of theorem 2.4.

Recall that a t-qubit error channel is one for which there exists a Kraus decomposition where each Kraus operator is a sum of weight t linear operators. By theorem 2.4 and the argument above for the dephasing channel, any code that corrects arbitrary t-qubit linear errors will therefore also correct arbitrary t-qubit error channels.

What about independent channels?

Theorem 2.6. *Let $\mathcal{I}$ be the 1-qubit identity channel and $\mathcal{E} = \otimes_{i=1}^n \mathcal{E}_i$ be an n-qubit independent channel, with $\|\mathcal{E}_i - \mathcal{I}\|_\diamond < \epsilon \leq \frac{t+1}{n-t-1}$, and let U and $\mathcal{D}$ be the encoder and decoder for a QECC with n physical qubits that corrects t-qubit errors. Then*

$$\|\mathcal{D} \circ \mathcal{E} \circ U - \mathcal{I}\|_\diamond < 2\binom{n}{t+1}(e\epsilon)^{t+1}. \tag{2.44}$$

In other words, a QECC that can correct t-qubit errors also corrects small independent error channels up to a very good approximation. A similar statement immediately follows from theorem 1.1. This should thus be viewed as a specialization of theorem 1.1 with better constant factors and a much easier proof.

Proof. The strategy is straightforward: Expand $\mathcal{E}$ as a sum of a t-qubit error map (not normalized) and an additional small term. The t-qubit error map is corrected by the code due to linearity, and then we just have to bound the size of the additional term to prove the theorem.

Since $\|\mathcal{E}_i - \mathcal{I}\|_\diamond < \epsilon$, we can write $\mathcal{E}_i = \mathcal{I} + \delta\mathcal{E}_i$, with $\|\delta\mathcal{E}_i\|_\diamond < \epsilon$. Now,

$$\mathcal{E} = \mathcal{F} + \mathcal{G}, \tag{2.45}$$

where

$$\mathcal{F} = \sum_{r=0}^{t} \sum_{|S|=r} \bigotimes_{i \in S} \delta\mathcal{E}_i \tag{2.46}$$

$$\mathcal{G} = \sum_{r=t+1}^{n} \sum_{|S|=r} \bigotimes_{i \in S} \delta\mathcal{E}_i \tag{2.47}$$

are the sums over all tensor factors of $\leq t$ and $> t$ of the $\delta\mathcal{E}_i$s, respectively. (The tensor with $\mathcal{I}$ on other qubits is implicit.)

The first term $\mathcal{F}$ is a sum of terms acting on at most t qubits. Each term in the sum is not a completely positive map. In fact, $\delta\mathcal{E}_i$ is not even positive. However, $\delta\mathcal{E}_i$ is a difference of two completely positive maps, and therefore

$$\mathcal{F}(\rho) = \sum_k \alpha_k A_k \rho A_k^\dagger, \tag{2.48}$$

where the α_k coefficients can be either positive or negative real numbers and each A_k acts on at most k qubits. By theorem 2.4 (and the linearity of density matrices, as for the dephasing channel example), the QECC corrects $\mathcal{F}$.

By proposition 2.2, the scaling factor $c(E, |\psi\rangle)$ in definition 2.1, the definition of a QECC, does not depend on $|\psi\rangle$, which implies that

$$\mathcal{D} \circ \mathcal{F} \circ U = c\mathcal{I} \tag{2.49}$$

for some constant c.

Meanwhile,

$$\|\mathcal{G}\|_\diamond \leq \sum_{r=t+1}^{n} \sum_{|S|=r} \prod_{i \in S} \|\delta\mathcal{E}_i\|_\diamond \tag{2.50}$$

$$\leq \sum_{r=t+1}^{n} \binom{n}{r} \epsilon^r \tag{2.51}$$

$$\leq \binom{n}{t+1} (e\epsilon)^{t+1} = \delta \tag{2.52}$$

by lemma 1.2.

Now, $\mathcal{E}$, $\mathcal{D}$, and U are CPTP maps, so $\|\mathcal{D} \circ \mathcal{E} \circ U\|_\diamond = 1$. Thus,

$$1 - \delta \leq \|\mathcal{D} \circ \mathcal{E} \circ U\|_\diamond - \|\mathcal{D} \circ \mathcal{G} \circ U\|_\diamond \leq \|\mathcal{D} \circ \mathcal{F} \circ U\|_\diamond = \|c\mathcal{I}\|_\diamond = c \leq \|\mathcal{D} \circ \mathcal{E} \circ U\|_\diamond + \|\mathcal{D} \circ \mathcal{G} \circ U\|_\diamond \leq 1 + \delta. \tag{2.53}$$

That is, $|1 - c| \leq \delta$. Therefore,

$$\|\mathcal{D} \circ \mathcal{E} \circ U - \mathcal{I}\|_\diamond = \|\mathcal{D} \circ \mathcal{F} \circ U + \mathcal{D} \circ \mathcal{G} \circ U - \mathcal{I}\|_\diamond \tag{2.54}$$

$$= \|(c-1)\mathcal{I} + \mathcal{D} \circ \mathcal{G} \circ U\|_\diamond \tag{2.55}$$

$$\leq |c - 1| + \|\mathcal{G}\|_\diamond \tag{2.56}$$

$$\leq 2\delta. \tag{2.57}$$

□

2.5 The Quantum Error Correction Conditions

2.5.1 Sufficient Conditions (Non-degenerate Orthogonal Coding)

Recall the argument that we used to show that the nine-qubit code could correct phase errors. Very little about it was specific to the nine-qubit code. We said that phase errors generated subspaces which were orthogonal to the code space and to each other, and that therefore there was a measurement that distinguished

them. We can use this same argument to give a general sufficient condition to have a QECC: Suppose that $Q \subseteq \mathcal{H}_N$ is the code space, and let $E(Q)$ be the subspace formed by acting on Q with E. Then we want that for any pair of errors $E_1, E_2 \in \mathcal{E}$, the subspace $E_1(Q)$ is orthogonal to $E_2(Q)$. Then we will be able to make a measurement that identifies which subspace we are in and therefore identifies the error.

To be sure we have a QECC, we need a little bit more. After identifying the error, we need to be able to reverse it. That is only possible if $E|_Q$ is a unitary map from Q to $E(Q)$. $E|_Q$ is unitary iff $\langle\psi|E^\dagger E|\phi\rangle = \langle\psi|\phi\rangle$ for all $|\psi\rangle, |\phi\rangle \in Q$. Thus, we get the sufficient condition that $(Q, \mathcal{E})$ is a QECC if $\forall|\psi\rangle, |\phi\rangle \in Q$, $\forall E_a, E_b \in \mathcal{E}$,

$$\langle\psi|E_a^\dagger E_b|\phi\rangle = \delta_{ab}\langle\psi|\phi\rangle. \tag{2.58}$$

A code that satisfies this condition could be called a *non-degenerate orthogonal code*, although the term "orthogonal code" is not widely used.

2.5.2 The QECC Conditions

However, by looking at the nine-qubit code, we can already see that equation (2.58) is not a necessary condition. If $E_1 = Z_1$ and $E_2 = Z_2$, then equation (2.58) fails since $E_1|\overline{\psi}\rangle = E_2|\overline{\psi}\rangle$. We need to generalize slightly:

Theorem 2.7 (QECC Conditions)**.** *$(Q, \mathcal{E})$ is a QECC iff $\forall|\psi\rangle, |\phi\rangle \in Q$, $\forall E_a, E_b \in \mathcal{E}$,*

$$\langle\psi|E_a^\dagger E_b|\phi\rangle = C_{ab}\langle\psi|\phi\rangle. \tag{2.59}$$

Note that C_{ab} does not depend on $|\psi\rangle$ or $|\phi\rangle$.

Proof. $\Leftarrow$: Recalling theorem 2.4, we might as well pick a useful spanning set for $\mathcal{E}$ and restrict attention to that spanning set. Taking the adjoint of equation (2.59) and putting in $|\phi\rangle = |\psi\rangle$, we find that $C_{ab}^\dagger = C_{ba}^*$, i.e., the matrix C is Hermitian. Therefore C_{ab} is diagonalizable, and by choosing an appropriate spanning set $\{F_a\}$ for $\mathcal{E}$ we can actually diagonalize C_{ab}. (Note that it is not necessarily true that $F_a \in \mathcal{E}$, but that is not particularly relevant, it just means that the original set $\mathcal{E}$ is smaller than the maximal set of possible errors the code can correct.)

We have

$$\langle\psi|F_a^\dagger F_b|\phi\rangle = d_a\delta_{ab}\langle\psi|\phi\rangle. \tag{2.60}$$

This is almost equation (2.58), but the coefficient d_a can be different from 1. (d_a is an eigenvalue of C_{ab}, so it must be real.) However, it is still true that the different subspaces $F(Q)$ are orthogonal to each other, so we can make a measurement that determines which error F_a we have. It is not true that $F_a|_Q$ must be unitary, but if d_a is nonzero, $F_a|_Q$ can be written as a unitary followed by a uniform rescaling. The decoding then gives the original state rescaled by d_a. This is allowed by definition 2.1.

If d_a is zero, then $F_a|_Q$ cannot be inverted, since there is no state left. Formally, we are still OK, since in definition 2.1, we would just get that $c(F_a, |\psi\rangle) = 0$. Physically, this means that the error F_a *never occurs*. It has probability proportional to $\langle\psi|F_a^\dagger F_a|\psi\rangle$, which is 0.

Therefore, equation (2.59) gives sufficient conditions to have a QECC.

$\Rightarrow$: Recall that by proposition 2.2, $c(E, |\psi\rangle)$ and $|A(E, |\psi\rangle)\rangle$ don't depend on $|\psi\rangle$. More generally, if we purify the decoder to V, we get a unitary transformation, which preserves inner products:

$$\langle\overline{\psi}|E_a^\dagger E_b|\overline{\phi}\rangle = (\langle\overline{\psi}| \otimes \langle 0|)E_a^\dagger V^\dagger V E_b(|\overline{\phi}\rangle \otimes |0\rangle) \tag{2.61}$$

$$= \sqrt{c(E_a)c(E_b)}\,(\langle\psi| \otimes \langle A(E_a)|)(|\phi\rangle \otimes |A(E_b)\rangle) \tag{2.62}$$

$$= \sqrt{c(E_a)c(E_b)}\,\langle A(E_a)|A(E_b)\rangle\langle\psi|\phi\rangle. \tag{2.63}$$

This is equation (2.59) with $C_{ab} = \sqrt{c(E_a)c(E_b)}\langle A(E_a)|A(E_b)\rangle$.

□

2.5.3 Degenerate Codes

The difference between equation (2.58) and equation (2.59) consists in the fact that C_{ab} might not be δ_{ab}. When C_{ab} has maximum rank, this difference has no deep meaning, since we have just made a bad choice of basis errors. As in the proof of theorem 2.7, we can choose a different set of errors with the same span to diagonalize C_{ab}, and we can even rescale to make $C_{ab} = \delta_{ab}$.

When C_{ab} has non-maximal rank, we cannot do this. If the error set $\mathcal{E}$ is not linearly independent, C_{ab} cannot have maximal rank, since a linearly dependent set of errors will produce a linearly dependent set of rows in C_{ab}. The interesting case is when $\mathcal{E}$ is linearly independent. It is possible then to still have a QECC with non-maximal rank for C_{ab}. This is known as a degenerate code.

Definition 2.2. Suppose $(U, \mathcal{E})$ is a QECC and $\mathcal{E}$ is linearly independent. Then the code is *degenerate* if $\text{rank}(C_{ab}) < |\mathcal{E}|$. A code $(U, \mathcal{E})$ (for linearly dependent $\mathcal{E}$) is degenerate if $(U, \mathcal{E}')$ is degenerate, where $\mathcal{E}'$ is a minimal spanning set for $\mathcal{E}$. If a code is not degenerate, it is *non-degenerate*.

We have already seen an example of a degenerate code: the nine-qubit code. The essence of degeneracy is that different errors will produce the same result (or at least linearly dependent results) when acting on a codeword. In the nine-qubit code, any two Z errors acting on the same set of 3 qubits will produce the same result. For instance,

$$Z_1|\overline{0}\rangle = Z_2|\overline{0}\rangle = \frac{1}{2\sqrt{2}}(|000\rangle - |111\rangle)(|000\rangle + |111\rangle)(|000\rangle + |111\rangle) \tag{2.64}$$

$$Z_1|\overline{1}\rangle = Z_2|\overline{1}\rangle = \frac{1}{2\sqrt{2}}(|000\rangle + |111\rangle)(|000\rangle - |111\rangle)(|000\rangle - |111\rangle). \tag{2.65}$$

2.5.4 Distance

The most common situation when designing a QECC is to let the set of possible errors be all t-qubit errors. It is therefore worth examining theorem 2.7 more explicitly for a t-error correcting code. When E_a and E_b are both weight t errors, then $E_a^\dagger E_b$ is a weight $2t$ error.

Definition 2.3. Let $C \subseteq \mathcal{H}_N$ be a QECC. The *distance* of C is the minimum weight d of an error F such that

$$\langle\psi|F|\phi\rangle \neq c(F)\langle\psi|\phi\rangle, \tag{2.66}$$

where $|\psi\rangle$ and $|\phi\rangle$ run over all possible pairs of codewords of C.

Note that the notion of distance implies that the Hilbert space is broken up into qubits because weight is defined in terms of qubits. Naturally, we can easily generalize the distance to work with a code over qudits of dimension q by defining weight in terms of the number of qudits in the support of an operator.

We get the following corollary of theorem 2.7:

Corollary 2.8. *A distance d code corrects $\lfloor (d-1)/2 \rfloor$ errors.*

Inverting this formula, we find that to correct t errors, a code needs distance $2t + 1$. If the distance is even, the extra point is "wasted" for this application, but as we shall see in a moment, the extra point of distance can be helpful in alternative applications.

The three central properties of a QECC are the size of the logical subspace, the number of physical qubits, and the distance, so there is a notation encapsulating those properties.

Notation 2.4. A QECC which encodes $\mathcal{H}_K$ into $\mathcal{H}_{2^n}$ and has distance d is denoted as an $((n, K, d))$ code. If the physical Hilbert space is n qudits each of dimension q, it is an $((n, K, d))_q$ code. If the distance is unknown or irrelevant, it is an $((n, K))$ code or an $((n, K))_q$ code.

This notation is derived from an analogous one for classical error-correcting codes, which will be introduced in chapter 4. The classical notation uses single parentheses, and the double parenthesis indicates that we have a quantum error-correcting code.

Sometimes when we discuss a code we consider it to have a distance less than its true distance or to correct fewer errors than the maximum number it can correct. This is just a convenience indicating that we are ignoring some of the error-correcting capability of the code. However, the true distance of a QECC does give us the tightest estimate of the number of errors it can correct — that is, the converse of the corollary also holds. In order to get the definition of distance, we need F to run over all errors of weight $< d$, but the QECC conditions applied to a t-error correcting code only gives us equation (2.66) for errors of the form $E_a^\dagger E_b$, which does not include all possible weight $2t$ errors. However, equation (2.66) is linear in F, so it is sufficient to check the formula for a basis of the set of weight $2t$ errors, and the set of errors of the form $E_a^\dagger E_b$ does include such a basis.

By similar reasoning, you get the same notion of distance if you alter the definition of distance to use all d-qubit errors or just weight d tensor products of I, X, Y, and Z.

When describing a code without an explicit set of correctable errors, I said in the definition that you are supposed to determine the set of correctable errors by context. Typically, we will choose the error set to be the set of t-qubit errors, which makes the distance of a code one of its most critical properties. When the error set is given implicitly, we can define degeneracy in terms of the code's capability as a t-error correcting code.

Definition 2.5. Let $Q \subseteq \mathcal{H}_N$ be a QECC with distance $d = 2t + 1$. Then Q is *degenerate* if it is degenerate when the set of correctable errors is all t-qubit errors.

If the distance is even, $2t+2$, and there is no more structure, it makes the most sense to define degeneracy by also considering the set of correctable errors to be t-qubit errors, but for certain families of codes (such as stabilizer codes, discussed in chapter 3), we can do better.

2.5.5 Quantum Error Detection and Erasure Errors

The distance is useful information about a QECC partially because it tells you how many errors the code can correct, but also because it encapsulates some additional error-correction-related properties of the code. In particular, the distance also tells you about the code's ability to detect errors without correcting them and about the code's ability to correct erasure errors. These applications can take advantage of even distance codes, whereas correcting general t-qubit errors only needs odd distance codes.

Definition 2.6. An encoder U (defined as for a QECC) and a set of detectable errors $\mathcal{E}$ form a *quantum error-detecting code* if they have the following property: Let Π be the projector on the code space. Then $\Pi E|\psi\rangle = c(E, |\psi\rangle)|\psi\rangle$, for all $E \in \mathcal{E}$ and all codewords $|\psi\rangle$. The code space is defined as the image of U, as for a QECC, and we frequently refer to the code space as defining the error-detecting code instead of the encoder.

Based on the definition of an error-detecting code, the measurement $(\Pi, I - \Pi)$ will either project us back on the original state (with some probability $|c(E, |\psi\rangle)|^2$) or will identify that an error occurred. This condition can be easily rewritten in similar terms to the QECC conditions:

Theorem 2.9. *$(U, \mathcal{E})$ is a quantum error-detecting code iff*

$$\langle\psi|E|\phi\rangle = c(E)\langle\psi|\phi\rangle \tag{2.67}$$

for all codewords $|\psi\rangle$ and $|\phi\rangle$ and all $E \in \mathcal{E}$. A code with distance d detects arbitrary $(d-1)$-qubit errors.

Based on this theorem, we can understand the distance of the code as the minimum number of qubits on which we can act to produce an undetectable error, i.e., an error with a component taking a codeword to a *different* codeword. (An error taking a codeword to itself is considered "detectable" by the definition.)

Proof. $\Leftarrow$: We can write $\Pi = \sum |\psi_i\rangle \langle\psi_i|$ where the sum runs over a basis $|\psi_i\rangle$ for the code space Q. We can then calculate $\Pi E|\psi\rangle$ using equation (2.67) to see that we have a quantum error-detecting code.

$\Rightarrow$: Equation (2.67) follows immediately from the definition of a quantum error-detecting code if we can prove that $c(E, |\psi\rangle)$ does not depend on $|\psi\rangle$. This can be done by considering a superposition $\alpha|\psi_1\rangle + \beta|\psi_2\rangle$, as in proposition 2.2.

The definition of distance then shows that a distance d code detects $d-1$ errors. □

While I have presented quantum error-correcting codes and quantum error-detecting codes as different things, clearly there is a very close connection. A code able to correct t errors will also be able to detect $2t$ errors, and vice-versa. Detecting errors and correcting errors are really just two different applications for the same code, and henceforth, I won't make a distinction between a code designed to correct errors and one designed to detect errors. Both will be referred to as QECCs.

There is no unique maximal set of correctable errors for a QECC, but there is a unique maximal set of detectable errors. The quantum error-correction conditions involve a product of two errors, and there may be more than one set of errors that will run over all possibilities for the product. However, equation (2.67) is only linear in the error, so we can define a unique set of detectable errors.

Definition 2.7. Given a subspace Q, its set of *detectable errors* is

$$\{E \text{ s.t. } \langle\psi|E|\phi\rangle = c(E)\langle\psi|\phi\rangle \,\forall |\psi\rangle, |\phi\rangle \in Q\}. \tag{2.68}$$

If $\mathcal{E}_C$ is the set of correctable errors and $\mathcal{E}_D$ is the set of detectable errors for a code, then the QECC conditions can be rephrased as $\mathcal{E}_C^2 \subseteq \mathcal{E}_D$.

A code's ability to correct erasure errors can be understood just by applying the regular QECC conditions. The interesting twist in this case is that we can apply the side classical information we have about the location of the errors to correct twice as many errors:

Theorem 2.10. *A QECC with distance d can correct $d-1$ erasure errors.*

Proof. The best way to think about an erasure-correcting code is as a set of QECCs which all have the same encoder. Each code is associated with a different error set, depending on where the erasure errors took place. Since we don't know when doing the encoding where the errors are, we have to use the same encoder in all cases. When *decoding*, however, we know where the errors are (although not what kind of errors they are), so we can choose a decoder based on the actual error set — all possible errors on the actual set of qubits erased.

Therefore we need a single code space that satisfies the QECC conditions for any error set of the form $\mathcal{E}_S$, which is the set of all possible errors with support on the set S of at most t qubits. But $\mathcal{E}_S^2 = \mathcal{E}_S$ since the product of two errors with support on S still has support on S. For a distance d code, the set of detectable errors includes all $(d-1)$-qubit errors, so when $t \leq d-1$, all sets $\mathcal{E}_S$ are subsets of the set of detectable errors. □

2.5.6 Alternate Forms of the Quantum Error Correction Conditions

There are a number of variants of the QECC conditions which are useful in different contexts. You have just seen one relating a correctable set of errors with the set of detectable errors for a code. A few more appear in the following proposition:

Proposition 2.11. *The following are equivalent to the QECC conditions given in theorem 2.7:*

1. *For all codewords $|\psi\rangle$, all pairs $E_a, E_b \in \mathcal{E}$,*

$$\langle\psi|E_a^\dagger E_b|\psi\rangle = \mathrm{tr}(|\psi\rangle\langle\psi|E_a^\dagger E_b) = C_{ab}. \tag{2.69}$$

2. *If $\mathrm{span}(\mathcal{E}) = \mathcal{E}$: For any pair of codewords $|\psi\rangle, |\phi\rangle$, any error $E \in \mathcal{E}$,*

$$\langle\psi|E^\dagger E|\phi\rangle = C(E)\langle\psi|\phi\rangle. \tag{2.70}$$

3. If $\mathrm{span}(\mathcal{E}) = \mathcal{E}$*: For any codeword* $|\psi\rangle$*, any error* $E \in \mathcal{E}$*,*

$$\mathrm{tr}(|\psi\rangle\langle\psi|E^\dagger E) = C(E). \tag{2.71}$$

4. Let $\{|\psi_i\rangle\}$ *be a basis for the code space. For any* i *and* j *and for any pair* $E_a, E_b \in \mathcal{E}$*,*

$$\langle\psi_i|E_a^\dagger E_b|\psi_j\rangle = C_{ab}\delta_{ij}. \tag{2.72}$$

Proof. The QECC conditions from equation (2.59) immediately imply all four of these variant conditions, so we only need to show the reverse direction.

To show that the standard QECC conditions follow from the first variant above, pick arbitrary $|\phi_1\rangle$ and $|\phi_2\rangle$ in the code space and consider $|\psi\rangle = \alpha|\phi_1\rangle + \beta|\phi_2\rangle$ for different values of α and β. $|\phi_1\rangle$ and $|\phi_2\rangle$ may not be orthogonal, so we have

$$1 = |\alpha|^2 + |\beta|^2 + 2\,\mathrm{Re}(\alpha^*\beta\langle\phi_1|\phi_2\rangle). \tag{2.73}$$

Now, $|\psi\rangle$ is also in the code space, and

$$\begin{aligned} C_{ab} &= \langle\psi|E_a^\dagger E_b|\psi\rangle & (2.74)\\ &= |\alpha|^2\langle\phi_1|E_a^\dagger E_b|\phi_1\rangle + |\beta|^2\langle\phi_2|E_a^\dagger E_b|\phi_2\rangle + \alpha^*\beta\langle\phi_1|E_a^\dagger E_b|\phi_2\rangle + \alpha\beta^*\langle\phi_2|E_a^\dagger E_b|\phi_1\rangle & (2.75)\\ &= (|\alpha|^2 + |\beta|^2)C_{ab} + (\alpha^*\beta\langle\phi_1|E_a^\dagger E_b|\phi_2\rangle + \alpha\beta^*\langle\phi_2|E_a^\dagger E_b|\phi_1\rangle). & (2.76) \end{aligned}$$

If we first plug in $\alpha = \beta = 1/\sqrt{2(1 + \mathrm{Re}\langle\phi_1|\phi_2\rangle)}$, we find

$$\langle\phi_1|E_a^\dagger E_b|\phi_2\rangle + \langle\phi_2|E_a^\dagger E_b|\phi_1\rangle = C_{ab}(\langle\phi_1|\phi_2\rangle + \langle\phi_2|\phi_1\rangle). \tag{2.77}$$

Plugging in $\alpha = -i\beta = 1/\sqrt{2(1 - \mathrm{Im}\langle\phi_1|\phi_2\rangle)}$, we get

$$\langle\phi_1|E_a^\dagger E_b|\phi_2\rangle - \langle\phi_2|E_a^\dagger E_b|\phi_1\rangle = C_{ab}(\langle\phi_1|\phi_2\rangle - \langle\phi_2|\phi_1\rangle). \tag{2.78}$$

Putting these two equations together, we find

$$\langle\phi_1|E_a^\dagger E_b|\phi_2\rangle = C_{ab}\langle\phi_1|\phi_2\rangle, \tag{2.79}$$

as desired.

The second and third variants use a similar argument for E_a and E_b. The proof of the fourth version is left as an exercise. □

Applying a similar argument to definition of the set of detectable errors, we find that an operator E is detectable iff $\mathrm{tr}(\rho E)$ does not depend on the codeword ρ. This has an interesting interpretation: it says that an operator is detectable if and only if measurement of that operator reveals no information about the logical state of the code. Certainly, if measuring an operator does reveal information about the logical state, it will create errors in the encoded state. What is perhaps surprising is that first, that some element of that error cannot be detected, and second, that revealing encoded information is the *only* thing that prevents an error from being detectable.

Another version of this insight appears when we apply the QECC conditions specifically to erasure errors.

Proposition 2.12. *Let* $\mathcal{E}$ *be a set of erasure errors, each one corresponding to a set of erased qubits (or qudits). Then* Q *corrects* $\mathcal{E}$ *iff* ρ_S *is the same for all logical states* $|\psi\rangle$ *whenever* $S \in \mathcal{E}$*. Here* S *is a subset of qubits that can be erased and* ρ_S *is the encoded state restricted to* S*.*

Note that the proposition just says that Q can correct for erasures on the set S iff the codeword on S has no information about the encoded state. Another interesting feature of this is that the ability to correct for erasures only refers to single sets, not pairs of sets, whereas more generally the QECC conditions refer to pairs of errors. This means that any QECC has a unique maximal set of erasure errors that it can correct, whereas, as I mentioned before, there is not a unique set of general errors that can be corrected — there is some tradeoff between correcting different kinds of errors.

Proof. As in the proof of theorem 2.10, we think of the code as a set of QECCs labelled after-the-fact by the set of erased qubits. We specialize to a single set $S \in \mathcal{E}$ and want to show that Q corrects erasures on S iff ρ_S is independent of the encoded state. Once we fix S, correcting erasures on S is equivalent to correcting arbitrary linear operators supported on S.

Let $E_a = |k\rangle\langle j|$ and $E_b = |k\rangle\langle i|$ where i, j, and k are bitstrings labelling basis vectors for the qubits just in S. We use variant 1 of the QECC conditions from proposition 2.11. Since E_a and E_b are always supported on S,

$$\begin{aligned} \operatorname{tr}(|\psi\rangle\langle\psi| E_a^\dagger E_b) &= \langle i|\rho_S|j\rangle\langle k|k\rangle && (2.80) \\ &= (\rho_S)_{ij}. && (2.81) \end{aligned}$$

As we let i and j vary over all possible basis states for S, equation (2.69) says that ρ_S does not depend on which codeword $|\psi\rangle$ we have, proving the forward direction of the proposition.

The reverse direction follows easily from variant 1 or variant 3 of proposition 2.11: If ρ_S is independent of $|\psi\rangle$, then taking the trace of ρ with operators on S will also give something independent of $|\psi\rangle$. □

2.6 What Makes a Good QECC?

Before we get any further in our discussion of quantum error-correcting codes, it is worth stopping to think about what we want out of a code. The first thing that comes to mind is that we want it to be good at correcting errors, so we want the set $\mathcal{E}$ of correctable errors to be large. When we're dealing with a t-error channel, we want the distance of the QECC to be large.

In order to correct errors, we have to add some extra qubits. For instance, to make the 9-qubit code, we had to add 8 extra qubits to encode 1. That 9 : 1 ratio seems pretty bad, so we'd like to find codes that use less overhead. In other words, we want k, the number of encoded qubits, to be large, and n, the number of physical qubits, to be small. Certainly we need $n > k$, but we'd like the *rate* k/n to be as close to 1 as possible. It's not clear a priori whether it's better to have a code with large k or small k, so we'd like good examples of codes for any value of k.

In general, if we fix t (the number of errors corrected) and let k get larger, we can make k/n larger too. However, this is rarely a sensible thing to do. As n gets larger, there are more opportunities for errors, so the expected number of errors gets larger. A better plan is to fix t/n when we vary n. In that scenario, there is a definite limit on the rate k/n that can be achieved, and we'd like to get as close to it as possible. We'd also like to know, theoretically, what the best possible rate is so that we know what to shoot for when designing codes. A family of codes that has both k/n and t/n as constants when n gets arbitrarily large is known as a *good* family of codes, and such code families are known.

The three parameters n, k, and d encapsulate the most interesting properties of a code, but not the only interesting ones. In part II, we'll talk about fault-tolerant quantum computation, and we'd like codes that are good for doing fault-tolerant protocols. We'd like a code that has a nice succinct classical description — describing even a single state on n qubits could require specifying 2^n amplitudes, and to specify a QECC we might need to list all 2^k states in a basis for the code subspace. Preferably, the n and k values of the code will be in the correct size range for the application we have in mind — we wouldn't want to have to add lots of extra logical qubits in order to use an otherwise nice code.

We'd also like the encoding and decoding operations to be implementable with a reasonable number of quantum gates. Frequently, when there is a short description of the code, the encoder can be performed with a small circuit. For instance, this is the case for the stabilizer codes which will be introduced in chapter 3. Ideally, the encoder would run in time $O(n)$ for a code with n physical qubits, and that is harder to achieve. (An arbitrary stabilizer code takes time $\Omega(n^2/\log n)$ to encode.) We might want additional properties, for instance that the encoding circuit can be implemented easily in a two-dimensional layout.

Decoding is a much more sticky problem. Even codes with a simple description may have a very difficult decoder. For the 9-qubit code, we had an error syndrome which identified the error. The error syndrome is not well-defined for all QECCs, but even if we restrict attention to codes with an error syndrome, we are faced with the *syndrome decoding problem* of determining the actual error given the error syndrome.

The syndrome decoding problem is NP-hard for most broad classes of codes. (This is true even for classical error-correcting codes.) When we come up with new codes, we'd like ones for which the decoding problem is not nearly as hard. Again, the ideal solution would be a code for which the syndrome decoding problem can be solved in linear time. Such codes exist, but they are somewhat rare.

This is a rather long wish list of properties we'd like out of a code. It's actually possible to satisfy many of them at the same time, but we don't currently know of any QECCs that are perfect in all respects. In other words, choosing a code is a matter of trade-offs. We'll want to use one code for some purposes and a different code for other purposes, depending on which property is most desirable for our current goal.

One thing to bear in mind when looking for a new code is that codes which look different may have pretty much the same properties. For instance, if you switch the first and second qubits of a code, you probably have a different subspace than you did before, but it doesn't really deserve the name of a new code. We capture this intuition with the notion of equivalent codes:

Definition 2.8. Two QECCs C and C' on n physical qubits are *equivalent* if C' can be produced from C by performing some set of single-qubit unitaries and by permuting the qubits.

The notion of equivalence can of course be extended to codes on a q^n-dimensional Hilbert space as well. Two equivalent codes have the same basic properties.

Proposition 2.13. *If C and C' are equivalent, then C and C' have the same number of logical qubits and the same distance.*

It's not necessarily true that C and C' correct exactly the same set of errors, since the local unitaries and permutations of the qubits can scramble up the errors. However, it's always true that an equivalence will map errors into errors of the same weight. It's also true that a fast decoding procedure for C will give a fast decoding procedure for C', so the codes don't differ in computational complexity either.

When you're looking for new codes, if you find a code that's equivalent to a known code, you probably shouldn't consider it to be new. It's not always straightforward to tell if two codes are equivalent, however. The safest thing is to find codes with new (hopefully better) parameters $((n, K, d))$ than preexisting codes.

Chapter 3

Will The Real Codeword Please Stand Still?: Stabilizer Codes

The treatment of QECCs in the last chapter was very general, but a bit unwieldy, particularly when it comes to finding new codes. We'd like a better method of discussing, manipulating, and finding QECCs, even if it comes at the cost of restricting somewhat the codes we can talk about. This chapter introduces the class of stabilizer codes, which have some nice properties and are much more tractable for most purposes than a general QECC. Stabilizer codes are built around a group of symmetries of the code that can distinguish between correct codewords and states with errors: The incorrect states will change under the action of the symmetry, while the correct ones will stay put.

3.1 The 9-Qubit Code Revisited

3.1.1 Error Syndrome Measurement for the 9-Qubit Code

The 9-qubit code is a stabilizer code, so I'll introduce the basic idea of a stabilizer code with some further examination of the nine-qubit code. Let's consider more carefully exactly what we measure when we measure the error syndrome for the nine qubit code. There are two parts to it: We measure each set of three qubits to see if one is different in the standard basis, and then we compare the three phases of the three sets of three qubits to see if one of the phases is different from the other two.

Recall that to determine if one of three qubits is different, we make two measurements. One measurement determines the parity of the first two qubits, and the other tells us the parity of the second and third qubits. Let us put that in a more quantum language: Determining the parity of two qubits is the same as measuring the eigenvalue of $Z \otimes Z$ on those qubits.

$$Z \otimes Z = \begin{pmatrix} +1 & 0 & 0 & 0 \\ 0 & -1 & 0 & 0 \\ 0 & 0 & -1 & 0 \\ 0 & 0 & 0 & +1 \end{pmatrix} \begin{matrix} 00 \text{ (even)} \\ 01 \text{ (odd)} \\ 10 \text{ (odd)} \\ 11 \text{ (even)} \end{matrix} \tag{3.1}$$

Eigenvalue $+1$ corresponds to even parity (syndrome bit 0) and eigenvalue -1 corresponds to odd parity (syndrome bit 1). Thus, within each set of three, to figure out if one qubit is different in the standard basis, we should measure $Z \otimes Z \otimes I$ and $Z \otimes I \otimes Z$.

We also need to determine whether two sets of three have the same phase or opposite phase. We'd like to phrase this as measurement of an eigenvalue. It should probably involve X, since the eigenvalues of X tell us the phase in $|0\rangle \pm |1\rangle$, but we actually have an entangled state. To determine the phase of $|000\rangle \pm |111\rangle$ we should instead measure the eigenvalue of $X \otimes X \otimes X$. To determine whether two sets of three have the same phase or opposite phase, we should thus measure the eigenvalue of the tensor product of six X's on

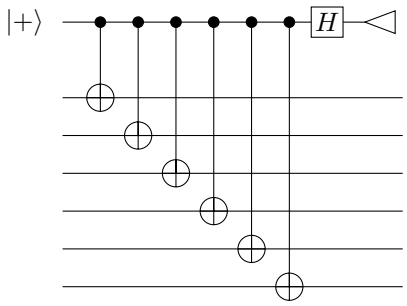

Figure 3.1: a) A non-fault-tolerant measurement procedure for $X \otimes X \otimes X \otimes X \otimes X \otimes X$.

Z	Z							
	Z	Z						
			Z	Z				
				Z	Z			
						Z	Z	
							Z	Z
X	X	X	X	X	X			
			X	X	X	X	X	X

Table 3.1: The generators of the stabilizer for the nine-qubit code. Blank spaces represent I operators.

the six relevant qubits. Remember, we want to learn whether the phases are the same or different without learning the actual value of each phase.

In case you're curious as to how one would actually measure this eigenvalue, you can look at figure 3.1. The question of how to measure the eigenvalue of a tensor product of arbitrary Paulis will be discussed in detail in section 12.1. Most of the focus of that section is on how to do the measurement fault-tolerantly, but it starts by discussing how to make a non-fault-tolerant measurement.

3.1.2 The Stabilizer for the 9-Qubit Code

The error syndrome for the 9 qubit code is 8 bits long: 2 bits for each set of three, identifying a bit flip error within that group of qubits, plus 2 additional bits telling us if and where a phase error has occurred. Putting together the list of all the operators whose eigenvalues give us those syndrome bits, we get the list in table 3.1. Each row corresponds to one operator, and the operator in column i indicates how that operator acts on qubit i. Blank spaces should be interpreted as I. Thus, the first row is the operator $Z \otimes Z \otimes I \otimes I \otimes I \otimes I \otimes I \otimes I \otimes I$. In other stabilizers, sometimes I will include the Is and sometimes omit them, whichever is most aesthetically appealing. (OK, maybe clarity plays a role too. And whimsy, definitely whimsy.)

As I mentioned in section 2.2, the choice of what qubits to measure for the error syndrome is not unique. For instance, we could have compared the parity of the first and third qubits. The parity would then be given by the eigenvalue of the operator $Z_1 Z_3$. Notice that $Z_1 Z_3$ is the product of the first two rows of table 3.1, and naturally the eigenvalue of the product will be the product of the eigenvalues. In general, we could use any product M of the operators listed in the table. The eigenvalue of M will tell us some information about the errors, but it won't be *new* information, since the eigenvalue of M can be determined from the eigenvalues of the operators used in the product.

Really, the operators we can measure to tell us about the error form a group, known as the *stabilizer* of the nine-qubit code. The eight operators listed in table 3.1 are the *generators* of the stabilizer. Choosing a

different way of defining the error syndrome of the code corresponds to choosing a different set of generators of the stabilizer.

Notice that we use tensor products of Zs to identify X (bit flip) errors, and we use tensor products of Xs to identify Z (phase flip) errors. The pattern here is that the errors anticommute with the stabilizer elements used to find them, and it is this property that makes stabilizers useful for discussing quantum error-correcting codes.

3.2 Pauli Group

I warned that you would be sick of the Pauli operators by the end of this book, and this chapter will get you started. Stabilizer codes are built around the Pauli operators, as you can perhaps see from the example of the nine-qubit code. It's therefore worth examining them in more detail.

Definition 3.1. The *Pauli group* P_n is composed of tensor products of I, X, Y, and Z on n qubits, with an overall phase of ± 1 or $\pm i$.

I will refer to elements of the n-qubit Pauli group as "Pauli operators," "Pauli errors," or just "Paulis" in the future; if I need to make a distinction between P_1 and P_n, I will do so explicitly.

The Pauli group, as its name suggests, is a group: It is closed under multiplication since $Y = iXZ$, and similarly, the product of any two of X, Y, and Z is equal to the third with an overall factor of $\pm i$. In addition, $X^2 = Y^2 = Z^2 = I$. Any tensor product of Paulis is its own inverse, i.e., it squares to I, and if there is an overall phase factor of $\pm i$, it instead squares to $-I$. (I use the one-qubit I and the n-qubit identity I interchangeably.)

The one-qubit Paulis anticommute with each other

$$XZ = -ZX \tag{3.2}$$

$$YZ = -ZY \tag{3.3}$$

$$XY = -YX. \tag{3.4}$$

Of course, I commutes with X, Y, and Z, and any Pauli commutes with itself. More general pairs (P, Q) of operators from P_n always either commute ($PQ = QP$) or anticommute ($PQ = -QP$). The difference from the single qubit case is that for P_1, only trivial pairs commute (when one Pauli is I or both are the same Pauli), but for $n > 1$, there are non-trivial commuting pairs. For instance, $X \otimes X$ commutes with $Z \otimes Z$. I will sometimes use the notation $[P, Q] = PQ - QP$ for the commutator and $\{P, Q\} = PQ + QP$ for the anticommutator.

P_n has 4^{n+1} elements since there are 4^n n-fold tensor products of I, X, Y, and Z, and 4 overall phases they could have. However, for many purposes, we can ignore the phase, giving us effectively 4^n distinct Paulis. If I wish to ignore the phase, I will refer to $\hat{\mathsf{P}}_n$. Thus, $\hat{\mathsf{P}}_n \cong \mathsf{P}_n / \{I, iI, -I, -iI\}$. The elements of $\hat{\mathsf{P}}_n$ are sets of 4 operators of the form $\{\pm P, \pm iP\}$, with P some n-qubit tensor product of I, X, Y, and Z. Each element of $\hat{\mathsf{P}}_n$ can thus be associated with an unsigned Pauli operator P, and it is more convenient by far to refer to elements of $\hat{\mathsf{P}}_n$ in terms of the associated Pauli rather than as a set of Paulis with signs.

Definition 3.2. Let $P \in \mathsf{P}_n$. Then $\hat{P} \in \hat{\mathsf{P}}_n$ is the element of $\hat{\mathsf{P}}_n$ corresponding to P. Similarly, if S is a subset of P_n, then $\hat{S}$ is the subset of $\hat{\mathsf{P}}_n$ consisting of $\hat{P}$ for all $P \in S$.

I have introduced these conventions separating P_n and $\hat{\mathsf{P}}_n$ in order to be mathematically precise. In some cases, we need to work with elements or subsets of P_n, and in some cases, to get the details completely correct, it is better to work with $\hat{\mathsf{P}}_n$. However, almost always, the distinction between the two is a small technical detail. My advice is to ignore the difference between P_n and $\hat{\mathsf{P}}_n$ unless you get confused.

Any Pauli operator has eigenvalues $+1$ and -1. Each eigenspace is exactly half the Hilbert space. That is, the $+1$ and -1 eigenspaces have dimension 2^{n-1}.

One feature of the Pauli group already mentioned is that P_n spans the space of all n-qubit linear operations. The weight t Paulis span the set of t-qubit errors. As discussed in corollary 2.5, this property means that to have a t-error correcting QECC, it suffices to correct all weight t Pauli operators.

This section has only discussed the most basic properties of the Pauli group. It has many more useful properties, some of which we will encounter in future sections of the book. Don't be surprised, however, if someday you are trying to solve a problem and discover a new property of the Paulis that is not mentioned in this book. In group theory, the Pauli group is an *extraspecial* group, and it truly deserves that name.

3.3 Stabilizer Codes

3.3.1 Definition and Basic Properties of the Stabilizer

Given a general quantum error-correcting code, we can define its stabilizer in more or less the same way as we did for the nine-qubit code:

Definition 3.3. Given a QECC $T \subseteq \mathcal{H}_{2^n}$, the *stabilizer* $\mathsf{S}(T)$ is

$$\mathsf{S}(T) = \{M \in \mathsf{P}_n | M|\psi\rangle = |\psi\rangle \ \forall |\psi\rangle \in T\}. \tag{3.5}$$

In other words, the stabilizer is composed of the Pauli operators for which all codewords are $+1$ eigenstates. In principle, one could define a stabilizer consisting of all unitaries which fix every codeword, but it turns out to be most useful to concentrate on the Pauli operators.

Proposition 3.1. *The stabilizer $\mathsf{S}(T)$ of a nonempty QECC T has three basic properties:*

a) $-I \notin \mathsf{S}(T)$

b) $\mathsf{S}(T)$ *is a group*

c) $\mathsf{S}(T)$ *is Abelian*

The property of being Abelian is the least obvious, but it makes sense since we want a set of operators which have the codewords as simultaneous eigenstates. Normally, for this to be possible, the operators should commute; in general, they only need to commute on a subspace, but for Pauli operators this is only possible if they truly commute.

Proof.

a) $-I$ has no $+1$ eigenstates, so it cannot be in the stabilizer.

b) If $M, N \in \mathsf{S}(T)$, it is clear that their product is also in $\mathsf{S}(T)$: For any $|\psi\rangle \in T$,

$$MN|\psi\rangle = M|\psi\rangle = |\psi\rangle. \tag{3.6}$$

c) By the argument for property 2, $MN|\psi\rangle = NM|\psi\rangle$ for any $|\psi\rangle \in T$, $M, N \in \mathsf{S}(T)$. Thus, $[M, N]$ annihilates the codewords. Paulis either commute or anticommute. If M and N anticommute, then $[M, N] = 2MN$, which is again an element of P_n, and therefore has no 0 eigenvalues. Therefore, the only option is that M and N commute.

□

It turns out to be most useful, when dealing with stabilizers, to work the other way around. We are given (or invent) a stabilizer with the properties we desire, and then use it deduce the code.

Definition 3.4. Let $\mathsf{S} \subseteq \mathsf{P}_n$ be an Abelian group, with $-I \notin \mathsf{S}$. Then define the code space corresponding to S by

$$\mathcal{T}(\mathsf{S}) = \{|\psi\rangle \text{ s.t. } M|\psi\rangle = |\psi\rangle \ \forall M \in \mathsf{S}\} \tag{3.7}$$

In general, $T \subseteq \mathcal{T}(\mathsf{S}(T))$, but when they are actually equal, the code has special properties.

X	Z	Z	X	I
I	X	Z	Z	X
X	I	X	Z	Z
Z	X	I	X	Z

Table 3.2: The generators for the five-qubit code.

Definition 3.5. If T is a QECC with $T = \mathcal{T}(\mathsf{S}(T))$, it is a *stabilizer code*.

Stabilizer codes are also sometimes known as *symplectic codes*, *additive codes*, or GF(4) *additive codes* for reasons that will be discussed in section 3.5 and section 5.2.

If we define a code from its stabilizer, then it is always a stabilizer code. In other words, a stabilizer code is one that could be defined just by giving its stabilizer. This is a consequence of the following proposition:

Proposition 3.2. $\mathsf{S} = \mathsf{S}(\mathcal{T}(\mathsf{S}))$.

I delay the proof until section 3.5 to keep you in suspense. Also, the proof will use a tool I won't introduce until then.

When dealing with stabilizer codes, I will frequently refer to the stabilizer S as the code instead of using the rather unwieldy phrase "stabilizer of the code $\mathcal{T}(\mathsf{S})$." When $M \in \mathsf{S}$, I will call M a "stabilizer element" or some variant of that phrase. However, many people seem to find that terminology unwieldy too, and just call M a "stabilizer." I do not approve of that usage, but I can't stop you if you want to say it.

Frequently we will want to pick a particular generating set $\{M_1, \ldots, M_r\}$ for the stabilizer, as we did for the nine-qubit code. In a minimal generating set no generator is a product of other generators. When we take a product of generators, the order does not matter since the generators commute. Also, note that since $-I \notin \mathsf{S}$, all elements of S must square to I, and any power of a generator greater than 1 can be reduced. Therefore, the elements of the stabilizer are uniquely determined by taking a product

$$M_{i_1,\ldots,i_r} = \prod_j M_j^{i_j}, \tag{3.8}$$

with $i_j \in \{0, 1\}$. This implies $|\mathsf{S}| = 2^r$.

Naturally, the set of generators is not unique, but when we do have a particular generating set to work with, I will talk about its elements as "stabilizer generators" or just "generators." When you describe a stabilizer, almost always the easiest way to do so is by listing one particular set of generators. The generators are enough to define the whole stabilizer, and it is much easier to list r generators than to list all 2^r elements of the stabilizer.

The code space of a stabilizer code is defined by requiring that all the eigenvalues be $+1$, but there is nothing magical about the $+1$ eigenstates. After all, a -1 eigenstate of M is a $+1$ eigenstate of $-M$, which is still in P_n. Given any set of generators $\{M_i\}$, we could define the code equally well by taking the -1 eigenstate, but switching M_i to $-M_i$. We *can't* arbitrarily change the eigenvalues associated with non-generators because the eigenvalues of the non-generators can be deduced from the eigenvalues of the generators. The generators are independent of each other, so we can pick their eigenvalues freely, but once we've done that, it defines the eigenvalues of all operators in the stabilizer. For more on this point, see section 3.5.

I'll conclude this subsection with a second example QECC to go with the nine-qubit code. This code has five physical qubits, and is, unsurprisingly, known as the "five-qubit code." The stabilizer for the five-qubit code is given in table 3.2. The code is cyclic: if you permute the qubits cyclically, you'll get the same stabilizer. You can almost see this from table 3.2. Shifting by one qubit for generators one through three gives you generators two through four. However, the fifth permutation $Z \otimes Z \otimes X \otimes I \otimes X$ is not one of the generators. Nevertheless, it is still in the stabilizer, as the product of all four generators given in the table. Even though it's not a generator, it still has the same status in the code as the four operators that are listed. Indeed, we could have chosen any four of these five operators as a set of generators without changing the code.

3.3.2 Projection Operator on a Stabilizer Code Subspace

In order to convert back from the representation of a stabilizer code in terms of its stabilizer to one as a subspace of a larger physical Hilbert space, we have definition 3.4, but it is sometimes helpful to have something more concrete. In particular, we'd like a projection operator that defines the subspace.

The codewords are supposed to be +1 eigenvectors of every element of the stabilizer, although it is sufficient to check that they are +1 eigenvectors of the stabilizer generators. The projector on the +1 eigenspace of generator M_i is $(I + M_i)/2$. Therefore, the projector on the code space of S is

$$\Pi_{\mathsf{S}} = \frac{1}{2^r} \prod_{i=1}^{r} (I + M_i), \tag{3.9}$$

when the stabilizer has r generators. Only states that are codewords — +1 eigenvectors of all generators — will avoid being annihilated by Π_{S}, and any codeword will be left alone by Π_{S}, as desired. Note that the order of the generators in the product does not matter since they all commute.

We can rewrite the projector Π_{S} in an interesting way by multiplying out the product. We get a sum of terms, each of which consists of a distinct product of the generators $\{M_i\}$. Based on equation (3.8), the products of the generators give us all elements of the stabilizer:

$$\Pi_{\mathsf{S}} = \frac{1}{2^r} \sum_{M \in \mathsf{S}} M. \tag{3.10}$$

We can come up with actual codewords for the code by applying Π_{S} to states in the physical Hilbert space and renormalizing. We might get 0, if the state we started with is orthogonal to the code space, but frequently we'll get a real codeword. For instance, for the five-qubit code, we can apply the projector to $|00000\rangle$ and $|11111\rangle$ to get two orthogonal codewords which can serve as a basis for the code space:

$$\begin{aligned} |\overline{0}\rangle = {} & |00000\rangle + |10010\rangle + |01001\rangle - |11011\rangle + |10100\rangle - |00110\rangle - |11101\rangle - |01111\rangle \\ & + |01010\rangle - |11000\rangle - |00011\rangle - |10001\rangle - |11110\rangle - |01100\rangle - |10111\rangle + |00101\rangle \end{aligned} \tag{3.11}$$

$$\begin{aligned} |\overline{1}\rangle = {} & |11111\rangle + |01101\rangle + |10110\rangle - |00100\rangle + |01011\rangle - |11001\rangle - |00010\rangle - |10000\rangle \\ & + |10101\rangle - |00111\rangle - |11100\rangle - |01110\rangle - |00001\rangle - |10011\rangle - |01000\rangle + |11010\rangle. \end{aligned} \tag{3.12}$$

I got these codewords by working my way through all 16 elements of the stabilizer for the five-qubit code (products of the generators given in table 3.2) and applying them to the starting states. You need to be careful of the signs involved, but otherwise the process is straightforward if tedious.

3.3.3 Distance and Size of a Stabilizer Code

When a stabilizer code is given either as a subspace or via the stabilizer, the number of physical qubits n is immediately obvious. The other two central properties (number k of logical qubits and distance d) are not so obvious, but can be deduced directly from the stabilizer.

Proposition 3.3. *If the stabilizer S on n physical qubits has $|\mathsf{S}| = 2^r$ (i.e., S has r generators), then $\mathcal{T}(\mathsf{S})$ has dimension 2^{n-r}. That is, there are $k = n - r$ logical qubits.*

Intuitively the result is easy to understand. The first generator of the stabilizer divides the Hilbert space into a +1 eigenspace and a −1 eigenspace of equal size. The codewords live in the +1 eigenspace. Each additional generator divides the subspace available for codewords in half again, so the total available dimension is 2^{n-r}. Non-generators do not divide the space since their eigenvalues can be derived by looking at the eigenvalues of the generators. This argument is not a proof, since we would need to show that the additional generators divide not just the full Hilbert space in half, but also all the smaller subspaces defined by the earlier generators. The actual proof is straightforward, but less intuitive:

Proof. The dimension of $\mathcal{T}(\mathsf{S})$ is equal to the trace of Π_{S}, the projection operator onto $\mathcal{T}(\mathsf{S})$. Now,

$$\operatorname{tr}\Pi_{\mathsf{S}} = \frac{1}{2^r}\sum_{M\in\mathsf{S}} \operatorname{tr} M. \tag{3.13}$$

However, $\operatorname{tr} P = 0$ for all Paulis P except for the identity. Thus,

$$\operatorname{tr}\Pi_{\mathsf{S}} = \frac{\operatorname{tr} I}{2^r} = \frac{2^n}{2^r}. \tag{3.14}$$

□

You can check the nine-qubit code in this formula: 9 physical qubits, 8 stabilizer generators, and 1 encoded qubit. For the five-qubit code, we have four generators, so again there should be 1 encoded qubit, with the basis codewords we saw above.

There's one special case which is not, strictly speaking, a QECC, but is interesting nonetheless. Yes, it's true, there are some things which are interesting other than quantum error correction. When the stabilizer has n generators on n qubits, proposition 3.3 would tell us we have 0 encoded qubits, which is a Hilbert space of dimension 1. That is, it is a single state, up to normalization.

Definition 3.6. A *stabilizer state* is the code space of a stabilizer with n generators on n qubits.

Since $r = n$ is the maximum number of generators you can have, a stabilizer state is an extreme limit of a QECC. Some constructions of QECCs will alter the number of encoded qubits from another code, and sometimes a stabilizer state can be the starting or ending point of such a construction. Also, stabilizer states are fairly common in the theory of quantum information, even discounting states arising from quantum error correction. For instance, the GHZ state $|000\rangle + |111\rangle$ and the Bell states $|00\rangle \pm |11\rangle$, $|01\rangle \pm |10\rangle$ are stabilizer states.

To understand how to deduce the distance of a stabilizer code, let us go back to the nine-qubit code and recall how it handled errors. The generators of the stabilizer were used to give us bits of the error syndrome. In particular, some generators were able to signal the presence of certain errors while missing other errors, but by looking at the full set of generators, we were able to identify all of the single-qubit errors.

I mentioned that the property responsible for determining whether a generator M is useful for an error E is anticommutation. Let us see how this works. Suppose $M \in \mathsf{S}$ and $E \in \mathsf{P}_n$ anticommutes with M. Then for any $|\psi\rangle \in \mathcal{T}(\mathsf{S})$,

$$M(E|\psi\rangle) = -EM|\psi\rangle = -E|\psi\rangle. \tag{3.15}$$

$|\psi\rangle$ was a $+1$ eigenvector of M — that is the definition of the code space — but $E|\psi\rangle$ is a -1 eigenvector.

Conversely, if E commutes with M then

$$M(E|\psi\rangle) = EM|\psi\rangle = E|\psi\rangle. \tag{3.16}$$

One advantage to dealing with the Pauli group is that these are the only choices. Either M and E commute or they anticommute. If we have an error that commutes with M, M retains a $+1$ eigenvalue, whereas if the error anticommutes with M, M's eigenvalue becomes -1, signaling that an error has occurred.

Definition 3.7. The *normalizer* $\mathsf{N}(\mathsf{S})$ of the stabilizer S is

$$\mathsf{N}(\mathsf{S}) = \{N \in \mathsf{P}_n | NM = MN \ \forall M \in \mathsf{S}\}. \tag{3.17}$$

If you know some group theory, you might recognize this as the definition of the *centralizer* of S (the set of things that commute with all elements of S) rather than the *normalizer* (the set of things that preserve S under conjugation), but because Paulis either commute or anticommute and $-I \notin \mathsf{S}$, they are the same thing for a stabilizer. I am choosing to call it the normalizer rather than the centralizer because the normalizer relates to logical operations, and this is an important function of $\mathsf{N}(\mathsf{S})$, as we shall see shortly in section 3.4.2.

Since the stabilizer is Abelian, $\mathsf{S} \subseteq \mathsf{N}(\mathsf{S})$ always. Indeed, $\mathsf{N}(\mathsf{S})$ also contains $-\mathsf{S}$ and $\pm i\mathsf{S}$. Since we worry about eigenvectors of S, changing the sign of an operator in the stabilizer is important, but $\mathsf{N}(\mathsf{S})$ is about errors, and global phase no longer matters. Therefore, we will usually want to work with $\hat{\mathsf{N}}(\mathsf{S})$.

The normalizer tells us which errors can be detected by the stabilizer code. If a Pauli E is outside the normalizer $\mathsf{N}(\mathsf{S})$, then $P|\psi\rangle$ has an eigenvalue -1 for some $M \in \mathsf{S}$, and thus is detected by measuring the eigenvalues of the stabilizer elements. Note that this is true for *any* codeword $|\psi\rangle$. Also note that it is sufficient to measure the generators of the stabilizer: If N commutes with M_1 and M_2, it also commutes with $M_1 M_2$. Thus, if N commutes with all generators of S, then $N \in \mathsf{N}(\mathsf{S})$.

When $E \in \mathsf{N}(\mathsf{S})$, then $E|\psi\rangle$ has eigenvalue $+1$ for all $M \in \mathsf{S}$, and therefore $E|\psi\rangle \in \mathcal{T}(\mathsf{S})$ for codewords $|\psi\rangle$. You might think that that means it is an undetectable error, but there is actually another class of Paulis that is "detectable" by definition 2.7. If $E \in \mathsf{S}$, then, while it's true that $E|\psi\rangle \in \mathcal{T}(\mathsf{S})$ for any codeword $|\psi\rangle$, it's also true that $E|\psi\rangle = |\psi\rangle$, so E is not actually an "error": it acts like the identity on the code space, leaving codewords unchanged. Ignoring global phases, we can say the same if $\hat{E} \in \hat{\mathsf{S}}$.

Putting this together, we get a characterization of the detectable errors for a stabilizer code.

Theorem 3.4. *The set of undetectable errors for a stabilizer code* S *is* $\hat{\mathsf{N}}(\mathsf{S}) \setminus \hat{\mathsf{S}}$. *The distance of* S *is* $\min\{\mathrm{wt}\, E | \hat{E} \in \hat{\mathsf{N}}(\mathsf{S}) \setminus \hat{\mathsf{S}}\}$.

The slanty line is a "set minus" operation. That is, $\hat{\mathsf{N}}(\mathsf{S}) \setminus \hat{\mathsf{S}}$ consists of those elements of $\hat{\mathsf{N}}(\mathsf{S})$ that are not in $\hat{\mathsf{S}}$.

Proof. We can prove the first statement directly from the definition of the set of detectable errors (definition 2.7). The second statement will then follow immediately from the definition of distance. We need to consider three cases for E. In cases 1 and 2, $|\psi\rangle$ and $|\phi\rangle$ are arbitrary codewords.

1. Case 1: $\hat{E} \in \hat{\mathsf{S}}$. Then for some choice of phase, $E \in \mathsf{S}$ and $E|\phi\rangle = |\phi\rangle$, so
$$\langle\psi|E|\phi\rangle = \langle\psi|\phi\rangle, \tag{3.18}$$
and $c(E) = 1$. $\hat{E}$ is detectable.

2. Case 2: $\hat{E} \notin \hat{\mathsf{N}(\mathsf{S})}$. Then $\exists M \in \mathsf{S}$ such that $\{M, E\} = 0$ (for any choice of phase of E), so $M(E|\phi\rangle) = -E|\phi\rangle$, as per equation (3.15). Then
$$\langle\psi|E|\phi\rangle = \langle\psi|MEM|\phi\rangle = -\langle\psi|E|\phi\rangle = 0, \tag{3.19}$$
since $M^2 = I$ for stabilizer elements. Thus $c(E) = 0$ and E is detectable.

3. Case 3: $\hat{E} \in \hat{\mathsf{N}}(\mathsf{S}) \setminus \hat{\mathsf{S}}$. Since $E \notin \mathsf{S}$, $\exists$ codeword $|\phi\rangle$ such that $E|\phi\rangle \neq |\phi\rangle$. Let $|\psi\rangle = E|\phi\rangle$. Since $E \in \mathsf{N}(\mathsf{S})$, $|\psi\rangle$ is also a codeword. However,
$$\langle\psi|E|\phi\rangle = 1 = \frac{1}{\langle\psi|\phi\rangle}\langle\psi|\phi\rangle, \tag{3.20}$$
whereas
$$\langle\phi|E|\phi\rangle = \langle\phi|\psi\rangle = (\langle\phi|\psi\rangle)\langle\phi|\phi\rangle. \tag{3.21}$$
Comparing equations (3.20) and (3.21) to definition 2.7 tells us that $\hat{E}$ is not detectable unless $|\langle\phi|\psi\rangle| = 1$, meaning $E|\phi\rangle = e^{i\theta}|\phi\rangle$. Furthermore, by equation (3.21), $\hat{E}$ is undetectable unless θ is the same for all $|\phi\rangle$. But that is not possible, since that would imply $\hat{E} \in \hat{\mathsf{S}}$.

□

The key point here is that when $E \in \hat{\mathsf{N}}(\mathsf{S}) \setminus \hat{\mathsf{S}}$, then E maps some codewords to *different* codewords. That's the essence of an undetectable error, because the code has no way of knowing if a codeword is the original encoding of the state or the result of the action of the error on a different codeword.

Moving now to correcting errors, we find

X	X	X	X
Z	Z	Z	Z

Table 3.3: The stabilizer for the four-qubit code.

Theorem 3.5. *The stabilizer code* S *corrects a set of errors* $\mathcal{E} \subseteq \mathsf{P}_n$ *iff* $\hat{E}^\dagger \hat{F} \notin \hat{\mathsf{N}}(\mathsf{S}) \setminus \hat{\mathsf{S}}$ *for all* $E, F \in \mathcal{E}$.

The theorem follows immediately from theorem 3.4 by comparing definition 2.7 with theorem 2.7. Recall that by the linearity of QECCs, and particularly corollary 2.5, it suffices to consider Pauli errors to understand the error-correcting capabilities of the code, at least when we are interested in correcting t-qubit errors.

For stabilizer codes, we have a slightly different notation than the more general $((n, K, d))$ notation that applies to arbitrary QECCs. Since the encoded subspace has a dimension that's always a power of 2, we write the code in terms of the number of encoded qubits k rather than the dimension $K = 2^k$ of the logical Hilbert space. We also use square brackets to indicate that we are dealing with a stabilizer code.

Notation 3.8. A stabilizer code with n physical qubits, k logical qubits, and distance d is denoted as an $[[n, k, d]]$ code. If the distance is unknown or irrelevant, it is an $[[n, k]]$ code.

We know that the nine-qubit code corrects a single-qubit error, so it must have distance at least 3. In fact, there are some 3-qubit Paulis (such as $X_1X_2X_3$) in $\hat{\mathsf{N}}(\mathsf{S}) \setminus \hat{\mathsf{S}}$ for the code, so the distance is exactly 3. Thus, the nine-qubit code is a $[[9, 1, 3]]$ code. The five-qubit code also turns out to have distance 3, so it is a $[[5, 1, 3]]$ code. Note that these codes are also $((9, 2, 3))$ and $((5, 2, 3))$ codes, but the more specific notation for a stabilizer code is generally used for them. As it happens, the five-qubit code is the smallest distance 3 code. You'll see the proof of that, as well as other techniques for proving limits on QECCs, in chapter 7.

You might wonder about distance 2 codes, since all the QECCs we have seen so far have distance 3. Distance 2 codes tend to be much simpler, but they can still be interesting. After all, a distance 2 code can detect one error or correct one erasure. The smallest distance 2 code is a $[[4, 2, 2]]$ code given in table 3.3. It is not hard to see that any one-qubit Pauli will anticommute with one or both of the two generators, but there are some two-qubit Paulis (e.g., $X \otimes X \otimes I \otimes I$) that commute with both. Thus, the code has distance 2.

3.3.4 Degeneracy and Stabilizer Codes

When is a stabilizer code degenerate? In the proof of theorem 3.4, we implicitly calculated the matrix C_{ab}. In particular, we found that $C_{ab} = 0$ if $E^\dagger F \notin \mathsf{N}(\mathsf{S})$ and $C_{ab} = 1$ if $E^\dagger F \in \mathsf{S}$. Any set of Paulis is linearly independent, so we just need to check if the resulting C_{ab} has maximum rank.

If $E_1^\dagger E_2 \in \mathsf{S}$, then $E_1^\dagger F \in \mathsf{S} \Leftrightarrow E_2^\dagger F \in \mathsf{S}$, so in this case the rows (or columns) of C_{ab} associated with E_1 and E_2 are the same. When $E_1^\dagger E_2 \notin \mathsf{S}$, then only one (or neither) of $E_1^\dagger F$ and $E_2^\dagger F$ can be in S, so in this case the rows associated with E_1 and E_2 cannot have 1s in the same column. Thus, we get the following proposition:

Proposition 3.6. *A stabilizer code* S *is degenerate for the error set* $\mathcal{E}$ *iff* $\exists E_1, E_2 \in \mathcal{E}$ *with* $E_1^\dagger E_2 \in \mathsf{S}$.

As with general QECCs, we can define degeneracy as a property of the code subspace without referring to specific set of errors by considering the distance. In the case of stabilizer codes, motivated by the above analysis, we can also sensibly define degeneracy when the code is used for error detection rather than error correction.

Definition 3.9. A stabilizer code S with distance d is *degenerate* if $\exists M \in \mathsf{S}$, $M \neq I$, with $\operatorname{wt} M < d$.

Note that this is slightly more general than the generic notion of a degenerate code. A stabilizer code with distance $d = 2t+2$ can be degenerate if S contains any elements of weight $2t+1$, but degeneracy wasn't defined for general non-stabilizer QECCs of even distance.

Looking once more at the stabilizer of the nine-qubit code (table 3.1), it is immediately obvious that it is a degenerate code. There are many generators of weight 2 and the code has distance 3. However, it is not always obvious from looking at the generators whether a stabilizer code is degenerate or not. It may be that the set of generators given to you all have high weight, but there is some product of the generators that does not. The five-qubit code is non-degenerate because all of the operators in the stabilizer have weight 4, but to see that you need to do some work.

3.4 Cosets and Error Syndromes

3.4.1 The Error Syndrome and the Stabilizer

For the nine-qubit code, we figured out the stabilizer by thinking about what we wanted to measure for the error syndrome. Each generator corresponded to a bit of the error syndrome. In fact, this is completely general:

Definition 3.10. Suppose S is a stabilizer code with generators $M_1, \ldots, M_r$ and $|\psi\rangle$ is an eigenvector (not necessarily with eigenvalue $+1$) of all $N \in \mathsf{S}$. The *error syndrome* of $|\psi\rangle$ is an r-bit string with ith bit 0 if $|\psi\rangle$ is a $+1$ eigenvector of M_i and ith bit 1 if $|\psi\rangle$ is a -1 eigenvector of M_i. The error syndrome $\sigma(E) : \mathsf{P}_n \to \mathbb{Z}_2^r$ is the error syndrome of $E|\phi\rangle$ for any codeword $|\phi\rangle$. When the stabilizer associated with a syndrome is in doubt, we will write $\sigma_{\mathsf{S}}(E)$.

If $P, Q \in \mathsf{P}_n$, then $c(P,Q) = 0$ if P and Q commute and $c(P,Q) = 1$ if they anticommute. ($c(P,Q) : \mathsf{P}_n \times \mathsf{P}_n \to \mathbb{Z}_2$.)

For a Pauli, bit i of the error syndrome is the same as $c(E, M_i)$. Note that the syndrome of E will be the same no matter what codeword $|\psi\rangle$ we choose to evaluate. This follows from equations (3.15) and (3.16) and means that Pauli errors map the code space, which is a subspace with $+1$ eigenvalue for all stabilizer elements, to a subspace that has a different set of eigenvalues, but for which each eigenvalue is still shared for all states in the subspace.

The subspaces derived this way are also error-correcting codes, but they are associated with different eigenvalues of the stabilizer. We can reinterpret them as traditional stabilizer codes (with all $+1$ eigenvalues) by replacing M_i with $-M_i$ whenever the ith syndrome bit is 1. Note that there are 2^r possible values of the error syndrome, and each subspace is isomorphic to the code space, which has dimension 2^k. When there are n physical qubits, $r = n - k$. Also, all the subspaces of different error syndromes are orthogonal to each other, since they have different eigenvalues. Thus, the full Hilbert space $\mathcal{H}_{2^n}$ decomposes as a direct sum of the 2^{n-k} subspaces associated with different syndromes.

Proposition 3.7. *The $c(P,Q)$ and $\sigma(E)$ functions have the following properties. In all cases, $+$ refers to binary addition.*

a) $c(P_1P_2, Q) = c(P_1, Q) + c(P_2, Q)$.

b) $c(P,Q) = c(Q,P)$.

c) $\sigma(EF) = \sigma(E) + \sigma(F)$.

These are straightforward to verify.

The various error syndromes are associated with different cosets of the normalizer in P_n.

Proposition 3.8. *Let $E, F \in \mathsf{P}_n$, and S be a stabilizer. Then F and E are in the same coset of $\mathsf{N}(\mathsf{S})$ iff E and F have the same error syndrome.*

Proof.

$\Rightarrow$: If E and F are in the same coset, then $F = EN$, with $N \in \mathsf{N}(\mathsf{S})$. Let $M \in \mathsf{S}$. Then $c(F,M) = c(E,M) + c(N,M)$. But $N \in \mathsf{N}(\mathsf{S})$, so $c(N,M) = 0$. Therefore the error syndromes of E and F are the same.

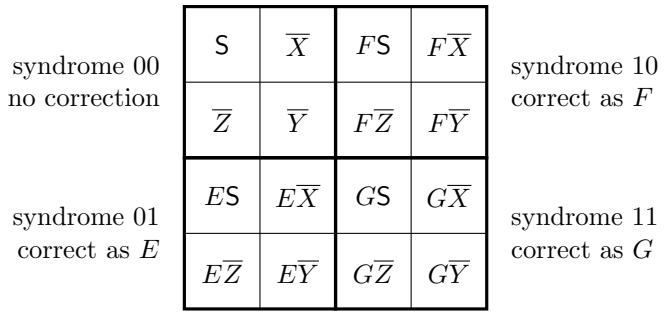

Figure 3.2: Costs of the normalizer and stabilizer in the Pauli group. The errors E, F, and G are chosen as canonical errors for the syndromes 01, 10, and 11.

$\Leftarrow$: The argument works the other way too: Let $N = E^\dagger F$. If the error syndromes of E and F are the same, then $c(N, M) = 0$ for all $M \in \mathsf{S}$. Therefore, $N \in \mathsf{N}(\mathsf{S})$.

□

In much the same way as the full Hilbert space can be written as a direct sum of subspaces associated to different syndromes, the whole Pauli group can be partitioned into cosets of $\mathsf{N}(\mathsf{S})$, each associated with a different syndrome. All the cosets are the same size and there are 2^r of them. As a consequence, we know the size of the normalizer:

Proposition 3.9. $|\mathsf{N}(\mathsf{S})| = 4 \cdot 2^{n+k}$ *when* S *has* n *physical qubits and* k *logical qubits.*

3.4.2 Cosets inside the Normalizer

We can similarly analyze the cosets of S in $\mathsf{N}(\mathsf{S})$.

Proposition 3.10. *Let* $N_1, N_2 \in \mathsf{N}(\mathsf{S})$. *Then* N_1 *and* N_2 *are in the same coset of* S *iff* $N_1|\psi\rangle = N_2|\psi\rangle$ *for all codewords* $|\psi\rangle$.

Proof. $N_1|\psi\rangle = N_2|\psi\rangle$ iff $|\psi\rangle$ is a $+1$ eigenstate of $M = N_1^\dagger N_2$. This is true for all codewords $|\psi\rangle$ iff $M \in \mathsf{S}$. However, N_1 and N_2 are in the same coset of S iff $N_2 = N_1 M$ with $M \in \mathsf{S}$. □

Recall that when $N \in \mathsf{N}(\mathsf{S})$, then $N|\psi\rangle$ is a codeword for any input codeword $|\psi\rangle$. Thus, N is a *logical operation*: it always maps codewords to (possibly different) codewords. Since the different representatives of a coset of S act the same way on codewords, they are all different realizations of the *same* logical operation. Thus, the set $\mathsf{N}(\mathsf{S})/\mathsf{S}$ is of particular interest, since it consists of the distinct logical operations performed by the normalizer. (S is a normal subgroup of $\mathsf{N}(\mathsf{S})$ by definition, so we can take this quotient without any difficulties.)

Theorem 3.11. *Let* S *be a stabilizer with* n *physical qubits and* k *logical qubits. Then* $\mathsf{N}(\mathsf{S})/\mathsf{S} \cong \mathsf{P}_k$.

The quotient group $\mathsf{N}(\mathsf{S})/\mathsf{S}$ can be taken to be the *logical Pauli group*, performing Paulis on the encoded qubits. The proof of theorem 3.11 will be in section 3.5.

We can also look at cosets of S in the full Pauli group P_n, as in figure 3.2. Each coset of $\mathsf{N}(\mathsf{S})$ breaks up into cosets of S, so we can associate each coset of S in P_n with an error syndrome, inherited from the coset of $\mathsf{N}(\mathsf{S})$ which it sits within. If we pick a particular representative E of the coset of $\mathsf{N}(\mathsf{S})$, then we can again identify the coset of S with a logical operation $\overline{P} \in \mathsf{N}(\mathsf{S})/\mathsf{S}$; the interpretation of the coset is then a combination of the logical operation $\overline{P}$ and the error E.

When we perform decoding on a stabilizer code, we do just this. For each error syndrome s, we assign a particular error E_s with $\sigma(E_s) = s$. If our syndrome measurement gives s, we assume the error actually

	X	Z	Z	X	I
	I	X	Z	Z	X
	X	I	X	Z	Z
	Z	X	I	X	Z
$\overline{X}$	X	X	X	X	X
$\overline{Z}$	Z	Z	Z	Z	Z

Table 3.4: The generators for the five-qubit code supplemented by representatives for logical Paulis.

	X	X	X	X
	Z	Z	Z	Z
$\overline{X}_1$	X	X	I	I
$\overline{X}_2$	X	I	X	I
$\overline{Z}_1$	I	Z	I	Z
$\overline{Z}_2$	I	I	Z	Z

Table 3.5: The stabilizer for the four-qubit code supplemented by representatives for logical Paulis.

was E_s and correct that. If the error was actually some E' with the same syndrome, we hope that E_s and E' are in the same coset of S. If not, there has been a logical Pauli error, the one associated with the actual coset of S in which E' lies.

Theorem 3.12. *If S is a non-degenerate stabilizer code, the error syndromes of all errors in the correctable error set $\mathcal{E}$ are distinct. If S is a degenerate code, E and F have the same error syndrome iff $\hat{E}^\dagger \hat{F} \in \hat{\mathsf{S}}$.*

Proof. By proposition 3.8, two errors $E \neq F \in \mathcal{E}$ have the same error syndrome iff they are in the same coset of $\mathsf{N}(\mathsf{S})$, meaning $E^\dagger F \in \mathsf{N}(\mathsf{S})$. But both errors are correctable, and by theorem 3.5, $\hat{E}^\dagger \hat{F} \notin \hat{\mathsf{N}}(\mathsf{S}) \setminus \hat{\mathsf{S}}$, so $E^\dagger F \in \mathsf{N}(\mathsf{S})$ iff $\hat{E}^\dagger \hat{F} \in \hat{\mathsf{S}}$. Thus, the error syndromes of two correctable errors E and F are the same iff $\hat{E}^\dagger \hat{F} \in \hat{\mathsf{S}}$. If this ever occurs, the code is degenerate. □

Applying this theorem is one way to see that the five-qubit code has distance 3 and is non-degenerate. There are 15 possible one-qubit errors (X, Y, Z on each of five qubits), plus the identity. There are 4 generators, so there are 16 error syndromes. If you list the syndromes of the 16 errors, you'll find that each one is unique. There are no syndromes left over, so the five-qubit code is known as a *perfect* code.

It is frequently helpful to also pick representatives of the cosets of S in the normalizer. These representatives don't have an interpretation in error correction, but they do give us concrete realizations of the logical Pauli group, which is helpful in fault-tolerant computation. For now, they are simply convenient computational tools. We don't actually need to pick representatives for every coset of S. Since $\mathsf{N}(\mathsf{S})/\mathsf{S}$ has a group structure itself, it is sufficient to pick representatives of generators of $\mathsf{N}(\mathsf{S})/\mathsf{S}$ and let the generators of other cosets be determined by multiplication. In other words, we can pick representatives for $\overline{X}_i$ and $\overline{Z}_i$ for $i = 1, \ldots, k$. (The subscript i here means the operator acts on the ith logical qubit, not the ith physical qubit.) I have done this for the five-qubit code and the four-qubit code in tables 3.4 and 3.5. Then, for instance, you can get $\overline{Y}$ for the five qubit code to be $i\overline{X}\overline{Z} = Y \otimes Y \otimes Y \otimes Y \otimes Y$.

You need to be careful when assigning cosets to elements of the logical Pauli group. Remember, $\mathsf{N}(\mathsf{S})/\mathsf{S} \cong \mathsf{P}_k$, and we'd like to realize this through a choice of representatives for the generating cosets. This means that the commutation and anticommutation relationships of the logical Paulis must be realized through the coset representatives. For instance, the coset representatives for $\overline{X}_i$ and $\overline{Z}_i$ must anticommute, while the coset representative of $\overline{X}_i$ must commute with the representatives for $\overline{X}_j$ or $\overline{Z}_j$ ($j \neq i$).

We could in principle do the same thing for $\mathsf{P}_n/\mathsf{N}(\mathsf{S})$, choosing representative errors only for the basis error syndromes and deducing the other errors by multiplication. However, in most cases, this is not a good thing to do. If our goal is to correct t errors for the maximum possible t, what we'd like to do instead is

choose the lowest weight error in each coset of $\mathsf{N}(\mathsf{S})$ as its representative. If we rely on multiplication to tell us the coset representatives, we'll frequently get errors of higher weight than necessary, and then there will be some lower weight errors that won't get corrected properly.

3.4.3 Maximum Likelihood Decoding

I'd like to depart briefly from the convention of most of this chapter, for which we had some fixed set $\mathcal{E}$ of errors that we'd like to correct, and we won't settle for anything less than correcting all of them. For the purposes of this subsection, assume we instead have some n-qubit Pauli channel, with $P \in \mathsf{P}_n$ occurring with probability p_P.

Given a stabilizer code S, we'd like to find a decoding procedure which minimizes the probability of error when the code goes through a Pauli channel. As discussed above, a decoding procedure can be understood as assigning to each error syndrome s a Pauli Q_s. When we measure s, we assume the actual error was Q_s and so perform the correction by inverting that error. If the actual error was something else, we may end up with the wrong state. In particular, if the true error P was in a different coset of S than Q_s was, we'll end up with some logical Pauli operation.

Thus, we can calculate the probability of logical error by summing over the cosets of S and $\mathsf{N}(\mathsf{S})$.

Proposition 3.13. *Let C_s be the coset of $\hat{\mathsf{N}}(\mathsf{S})$ with error syndrome s. $\hat{Q}_s\hat{\mathsf{S}}$ is the coset of $\hat{\mathsf{S}}$ containing the canonical error $\hat{Q}_s$ with syndrome s, so $\hat{Q}_s\hat{\mathsf{S}} \subseteq C_s$.*

a) The probability of error syndrome s (regardless of whether we correctly identify the error) is

$$p_s = \sum_{\hat{P} \in C_s} p_P. \tag{3.22}$$

b) The probability of having syndrome s and no logical error (i.e., error correction is successful) is

$$p_{s,\mathrm{OK}} = \sum_{\hat{P} \in \hat{Q}_s\hat{\mathsf{S}}} p_P. \tag{3.23}$$

c) The total probability of successfully decoding the state is

$$p_{\mathrm{OK}} = \sum_s \sum_{\hat{P} \in \hat{Q}_s\hat{\mathsf{S}}} p_P. \tag{3.24}$$

There is no interaction between the different error syndromes. If we change Q_s for one value of s, the only change to p_{OK} comes through the change to $p_{s,\mathrm{OK}}$. We can therefore treat each syndrome separately to maximize $p_{s,\mathrm{OK}}$. It doesn't matter which element of the coset $Q_s\mathsf{S}$ we choose, only which coset we choose within C_s.

In order to maximize the probability of successfully decoding, for each syndrome s, we should choose the coset $Q_s\mathsf{S}$ which maximizes $p_{s,\mathrm{OK}}$. As given by equation (3.23), that just means we look at each coset and add up the probability of all Paulis in the coset. Then we choose the coset with the highest value to be the representative coset for s.

The maximum likelihood decoder should be contrasted with the decoder that optimizes the distance, for which we choose the coset containing the lowest-weight Pauli. If we have an independent Pauli channel, the two procedures frequently, but not always, give the same answer: When the probability of error per qubit is small, a specific low-weight error will have higher probability than a specific high-weight error. However, a coset which contains one low-weight error and a bunch of high-weight errors may be less likely than a coset which contains a large number of medium-weight errors.

Getting the *exactly* optimal decoder, in the sense of always picking the most likely coset, is usually a computationally challenging task. Often, therefore, we are willing to settle for a good approximate decoder that gets p_{OK} to be close to the optimal value but with a much smaller computational cost.

In the Pauli group	Binary symplectic representation
$P \in \mathsf{P}_n$	$v_P = (x_P \vert z_P) \in \mathbb{Z}_2^n \times \mathbb{Z}_2^n$
Multiplication	Addition
$c(P, Q)$	$v_P \odot v_Q$
Phase	No equivalent
Stabilizer S	Weakly self-dual subspace S
Normalizer $\mathsf{N}(\mathsf{S})$	Dual subspace $\mathsf{S}^\perp$ (under $\odot$)
Minimal set of generators for S	Basis for S

Table 3.6: Equivalence between the Pauli group and its binary symplectic representation.

3.5 Binary Symplectic Representation

3.5.1 The Symplectic Representation of Paulis

One of the most useful techniques when dealing with stabilizer codes is to ignore the phases of Paulis and work with $\hat{\mathsf{P}}_n$ instead of P_n. We lose a couple of things by doing this. One is the ability to define the code space, which seems like a serious loss, but isn't really — as discussed above, the other eigenspaces of the stabilizer have the same error correction properties as the code space. The other missing thing is the ability to tell whether Paulis commute or anticommute, since $\hat{\mathsf{P}}_n$ is an Abelian group. This is a much greater loss, so we'll need to find a substitute.

The structure of $\hat{\mathsf{P}}_n$ is very straightforward. There are 4^n elements, and all pairs commute under multiplication. Therefore, $\hat{\mathsf{P}}_n \cong \mathbb{Z}_2^{2n}$. The group operation for $\mathbb{Z}_2^{2n}$ is usually written as addition. Conventionally, we choose to represent elements of $\hat{\mathsf{P}}_n$ as two n-bit binary vectors, one representing the "X" component and one representing the "Z" component.

Definition 3.11. The *binary symplectic representation* of $\hat{\mathsf{P}}_n$ is an isomorphism between $\hat{\mathsf{P}}_n$ and $\mathbb{Z}_2^n \times \mathbb{Z}_2^n$: $P \leftrightarrow v_P = (x_P|z_P)$. The ith component of x_P is 0 if P acts on qubit i as I or Z and 1 if P acts on qubit i as X or Y. The ith component of z_P is 0 if P acts on qubit i as I or X and 1 if P acts on qubit i as Y or Z. That is,

$$\begin{array}{c} I \\ X \\ Y \\ Z \end{array} \quad \longleftrightarrow \quad \begin{array}{c} (0|0) \\ (1|0) \\ (1|1) \\ (0|1) \end{array} \tag{3.25}$$

For instance, $X \otimes I \otimes Y \leftrightarrow (1\ 0\ 1|0\ 0\ 1)$. The "symplectic" refers to the substitute for commutativity/anticommutativity:

Definition 3.12. The *symplectic form* (or symplectic product) on $\mathbb{Z}_2^n \times \mathbb{Z}_2^n$ is $(x_1|z_1) \odot (x_2|z_2) = x_1 \cdot z_2 + x_2 \cdot z_1$, where the dot product is the usual scalar product in the vector space $\mathbb{Z}_2^n$ and addition is modulo 2.

Proposition 3.14. *$v_P \odot v_Q = c(P, Q)$ for $P, Q \in \mathsf{P}_n$.*

Proof. This can be checked exhaustively for a single qubit. For more than one qubit, note that both $\odot$ and $c(\cdot, \cdot)$ are equal to the parity of their single-qubit results. Therefore, since they are equal for single qubits, they are also equal for n qubits. □

Combining the symplectic form with the binary symplectic representation recovers the primary structure of the Pauli group. We can switch back and forth between the two representations to use whichever one is the most convenient at the moment. The conversion process is summarized in table 3.6. The one thing that is lost in the transition is the phase of a Pauli, so you must be careful whenever dealing with something that depends on that.

Some of the conversions need a little more explanation. There were three conditions for S to be a stabilizer: that $-I \notin \mathsf{S}$, that S is a group, and that S is Abelian. Since phase is lost when converting to

$$\begin{array}{c} \\ \\ \\ \\ \overline{X} \\ \overline{Z} \end{array} \left(\begin{array}{ccccc|ccccc} 1 & 0 & 0 & 1 & 0 & 0 & 1 & 1 & 0 & 0 \\ 0 & 1 & 0 & 0 & 1 & 0 & 0 & 1 & 1 & 0 \\ 1 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 1 & 1 \\ 0 & 1 & 0 & 1 & 0 & 1 & 0 & 0 & 0 & 1 \\ \hline 1 & 1 & 1 & 1 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 1 & 1 & 1 & 1 \end{array}\right)$$

Table 3.7: The binary symmetric representation of the stabilizer and logical Pauli operators for the five-qubit code.

the binary symplectic representation, the first condition is irrelevant. The second condition means that S becomes a linear subspace in $\mathbb{Z}_2^n \times \mathbb{Z}_2^n$, and the third means that $v \odot w = 0 \; \forall v, w \in \mathsf{S}$. This analysis and the conversion of the normalizer prompts the following definition:

Definition 3.13. Let V be a linear subspace of $\mathbb{Z}_2^n \times \mathbb{Z}_2^n$. The *dual* of V (with respect to $\odot$) is $V^\perp = \{w \in \mathbb{Z}_2^n \times \mathbb{Z}_2^n | w \odot v = 0 \; \forall v \in V\}$. A subspace is *self-dual* if $V^\perp = V$. A subspace is *weakly self-dual* if $V \subseteq V^\perp$.

Table 3.7 gives the five-qubit code in binary symplectic representation, including the logical $\overline{X}$ and $\overline{Z}$ operators.

Conversion can go the other way too. For instance, it is convenient to define a notion of a set of Paulis being independent based on the standard definition for binary vector spaces:

Definition 3.14. A set of Paulis $\{P_1, \ldots, P_m\}$ is *independent* iff the vectors $\{v_{P_1}, \ldots, v_{P_m}\}$ are linearly independent.

A set of Paulis is independent unless one of them is a product of others; this is equivalent to saying that a set of $2n$-bit binary vectors is independent unless one of them is a sum of others. Since the binary vector space is $2n$-dimensional, the maximum number of independent Paulis is $2n$, which is equal to the number of generators of the Pauli group.

3.5.2 Linear Algebra Lemma

The power of the binary symplectic representation is that we can use standard results from linear algebra to prove a number of properties of the Pauli group and stabilizers. In particular, we have the following lemma:

Lemma 3.15. *Let $\{P_1, \ldots, P_m\}$ be an independent set of Paulis on n qubits, and let s be an m-bit vector with components s_i. Then $\exists Q \in \mathsf{P}_n$ with $c(P_i, Q) = s_i$. In fact, there are 2^{2n-m} such Paulis.*

Proof. Converting to the binary symplectic representation gives us m linearly independent vectors v_i, and we wish to find vector w such that $v_i \odot w = s_i$. Each of these conditions is a linear equation, and since the vectors v_i are independent, the system of linear equations is non-singular. There are m equations in a $2n$-dimensional vector space, so the space of solutions has dimension $2n - m$. It is a binary vector space, so that corresponds to 2^{2n-m} solutions. □

An important consequence of the lemma is that it tells us about the decomposition of the full physical Hilbert space into subspaces associated with the different error syndromes. I've already discussed this decomposition, but the missing piece is given by the following corollary of lemma 3.15:

Proposition 3.16. *Let S be a stabilizer with generators $\{M_i\}$. Then for any error syndrome s, $\exists P \in \mathsf{P}_n$ with $\sigma(P) = s$.*

That is, not only is every Pauli outside $\mathsf{N}(\mathsf{S})$ associated with an error syndrome, but every possible error syndrome is associated with some Pauli.

3.5.3 Consequences of the Lemma

Now it's time for some of the proofs I owe you. There is proposition 3.2, which claims $\mathsf{S} = \mathsf{S}(\mathcal{T}(\mathsf{S}))$, and theorem 3.11, which says $\mathsf{N}(\mathsf{S})/\mathsf{S} \cong \mathsf{P}_k$.

First, here is the proof that $\mathsf{N}(\mathsf{S})/\mathsf{S} \cong \mathsf{P}_k$:

Proof of theorem 3.11. The most straightforward way to show this is by sequentially picking coset representatives for the cosets corresponding to $\overline{X}_i$ and $\overline{Z}_i$. The Pauli group P_k is determined, like any finitely-presented group, by its generators and relations between them. In the case of the Pauli group, the relations are

$$c(X_i, X_j) = 0 \tag{3.26}$$
$$c(Z_i, Z_j) = 0 \tag{3.27}$$
$$c(X_i, Z_j) = \delta_{ij}. \tag{3.28}$$

It is sufficient to consider only the generating cosets, since we can let the representatives of a product of cosets be the product of the representatives, as discussed in section 3.4.2. Provided we can choose $\overline{X}_i$ and $\overline{Z}_i$ with the correct commutation relations, that gives us an injective map of P_k into $\mathsf{N}(\mathsf{S})/\mathsf{S}$. We know that $|\mathsf{N}(\mathsf{S})| = 4 \cdot 2^{n+k} = |\mathsf{S}||\mathsf{P}_k|$, so an injection has to be isomorphism.

Suppose we've chosen some independent set of $\overline{X}_i$'s and $\overline{Z}_i$'s with the correct commutation relations, and we wish to choose one more. The new logical Pauli must be in $\mathsf{N}(\mathsf{S})$, so in particular, it must commute with all generators of the stabilizer. Second, it has a defined commutation relation with the already chosen logical Paulis. By lemma 3.15, there exists a Pauli that satisfies all of these constraints.

The only remaining thing to check is that the new logical Pauli can be chosen to be independent of the prior ones. To simplify the analysis, let us first pick all the $\overline{X}_i$'s, then the $\overline{Z}_i$'s. When we are picking $\overline{Z}_j$, it satisfies different commutation relations than all previously selected logical Paulis. In particular, $\overline{Z}_j$ anticommutes with $\overline{X}_j$, unlike all the previous logical Paulis and all the stabilizer generators. Thus, $\overline{Z}_j$ must be independent.

When we pick $\overline{X}_j$, this argument doesn't apply, since the new one will commute with all stabilizer generators and all previous logical Paulis. However, there are $n-k$ stabilizer generators and at most $k-1$ logical Paulis already chosen, for a total of at most $n-1$ constraints. By lemma 3.15, there are thus at least 2^{n+1} possible solutions. The group generated by the stabilizer and previous logical Paulis contains only 2^{n-1} operators, so there are possible choices for $\overline{X}_j$ that are independent of the previous choices.

□

Finally, we can prove proposition 3.2, showing that $\mathsf{S} = \mathsf{S}(\mathcal{T}(\mathsf{S}))$:

Proof of proposition 3.2. If $M \in \mathsf{S}$, then certainly $M|\psi\rangle = |\psi\rangle$ for any $|\psi\rangle \in \mathcal{T}(\mathsf{S})$, so $\mathsf{S} \subseteq \mathsf{S}(\mathcal{T}(\mathsf{S}))$. We need to show that if $N \notin \mathsf{S}$, then $N \notin \mathsf{S}(\mathcal{T}(\mathsf{S}))$. If $N \notin \mathsf{N}(\mathsf{S})$ then $N|\psi\rangle$ (for any codeword $|\psi\rangle$) has a different eigenvalue for some $M \in \mathsf{S}$, and is therefore orthogonal to the code space, which in turn means $N \notin \mathsf{S}(\mathcal{T}(\mathsf{S}))$.

If S has n generators, that is all we need. Otherwise, if $r < n$, we still need to show that if $N \in \mathsf{N}(\mathsf{S}) \setminus \mathsf{S}$, then $\exists|\psi\rangle \in \mathcal{T}(\mathsf{S})$ such that $N|\psi\rangle \neq |\psi\rangle$. By lemma 3.15, we can choose a $M \in \mathsf{N}(\mathsf{S})\setminus\mathsf{S}$ such that $\{M, N\} = 0$. Consider the modified stabilizer S' formed by adding M to S as a new generator. S' has $r+1$ generators, so its code space has dimension 2^{k-1} (by proposition 3.3). Since $k \geq 1$, there is at least one state $|\psi\rangle$ (up to normalization) in the code space of S'. $|\psi\rangle$ is left fixed by all elements of S', and in particular by the elements of $\mathsf{S} \subset \mathsf{S}'$. Thus, $|\psi\rangle \in \mathcal{T}(\mathsf{S})$. However, N anticommutes with M, which is in the stabilizer of $|\psi\rangle$. Therefore, $N|\psi\rangle$ has eigenvalue -1 for M, and in particular is orthogonal to $|\psi\rangle$. It follows that $N \notin \mathsf{S}(\mathcal{T}(\mathsf{S}))$, which proves the proposition.

□