PHY265 Lecture notes: Introducing Quantum Error Correction

A. C. Quillen

March 17, 2025

Contents

1 Error correction on a quantum computer **2**

2 Three bit Quantum error correction codes **3**
2.1 Correcting bit flip errors with a three bit code 3
2.1.1 Encoding is not equivalent to cloning 5
2.1.2 Bit flip error correction on a superposition state 6
2.1.3 A linear combination of bit flip errors 7
2.1.4 Bit flip error correction without measurement 8
2.1.5 Ancilla qubits must be reset in order to be reused 9
2.1.6 Projection operators and measurements for detecting bit-flip errors . 9
2.1.7 Phase errors are not detected by the 3-bit bit-flip error correction code 10
2.1.8 Subspaces 11
2.2 Correcting phase flip errors with a three qubit code 12

3 Shor's 9-bit code **13**
3.1 Shor's 9-bit code 14
3.2 Correcting all 1-qubit errors 17
3.3 Error Generated Subspaces 18
3.4 Correctable sets of errors 19

4 Theory of Quantum Error-Correction **20**
4.1 Encoding 21
4.2 Double bracket notation to describe codes 22

5 Stabilizer codes **22**
5.1 The Generalized Pauli Group 22
5.2 Weights and Quantum code distance 22

5.3 Stabilizer Groups . . . 23
5.4 Creating new stabilizers and stabilized subspaces via group conjugation . . 25
5.5 Stabilizer codes . . . 25
5.6 Normalizer and centralizers . . . 28
5.7 Examples of subspaces . . . 29
5.8 Syndromes using stabilizer generators for 3-bit codes and Shor's 9-bit code 29
5.9 Conditions for error correcting in Stablizer codes . . . 31
5.10 On degeneracy . . . 33
5.11 A non-generate 5 bit code . . . 33
5.12 Finding the encoding subspace from the stabilizer group . . . 34
5.13 The Quantum Hamming Bound . . . 35

6 CSS Codes **36**
6.1 Generator and parity matrices in classical codes . . . 36
6.2 The classical [7,4] Hamming code . . . 38
6.3 A perpendicular [7,3] Hamming code . . . 39
6.4 Constructing a CSS code . . . 39
6.5 Stabilizer group for Steane's 7 bit code . . . 40

7 Fault Tolerant Computing **41**
7.1 Operations on encoded states . . . 41
7.2 Fault tolerant NOT and Z gates for the 9 bit code . . . 44
7.3 Strategies for constructing fault tolerant operations . . . 45
7.4 Fault tolerant CNOT . . . 46

8 Topological Error Correcting Codes **47**
8.1 Surface Codes . . . 47
8.2 A 5 bit surface code . . . 48
8.3 Loops . . . 48
8.4 Toric Codes . . . 49
8.5 Q-dit toric codes . . . 49

9 Measuring errors on a quantum computer **51**
9.1 Quantum process tomography . . . 51
9.2 Randomized benchmarking . . . 52
9.3 The Clifford group . . . 54

1 Error correction on a quantum computer

Currently quantum computers are limited because entangled states are fragile. If one of the qubits decoheres, then the entanglement can be lost. If numerous gates are used, small

errors may add up. A goal of a quantum **error correction** scheme is to use extra qubits to correct errors.

Classically, suppose instead of sending a 1 we send 111 and instead of sending 0 we send 000. This is called a **repetition code**. If there is a single bit error, then the other two in the set of three can be used to detect and correct the error. An error in 2 or 3 bits of the three would be required for an error to remain undetected.

Are there analogies for quantum computing? One issue is that a qubit is not actually 1 or 0, rather it has a continuum of probabilities describing the likelihood that the system is in one of the states. In a superposition state of two states the relative phase between the two pieces is important.

Suppose you want the system to be $|0\rangle$ but actually an error has caused the state to be in a superposition; $\sqrt{1-\epsilon^2}\,|0\rangle + \epsilon\,|1\rangle$. You make a measurement. If you measure $|0\rangle$ you can ask, was there an error? If there was you no longer worry about it because the system is now in the $|0\rangle$ state. If you measure $|1\rangle$ you would know that there was an error and you would then call a Pauli-X operator to put the system back into $|0\rangle$. You can't do the measurement on the actual state as you would lose superposition. However if you can make a measurement in an encoded space that does not give you any information about the actual state, then you could correct the error.

A **quantum error-correcting code** can be viewed as a mapping of k qubits (a Hilbert space of dimension 2^k) into n qubits (a Hilbert space of dimension 2^n), where $n > k$. The k qubits are the "logical qubits" or "encoded qubits" that we wish to protect from error. The additional $n-k$ qubits allow us to store the k logical qubits in a redundant fashion, so that the encoded information is not easily damaged. If we make measurements that do not reveal any information about the original logical bits, then we can correct errors in the encoded space.

2 Three bit Quantum error correction codes

2.1 Correcting bit flip errors with a three bit code

Suppose we encode

$$\begin{aligned} |0\rangle \quad & \text{as} \quad |000\rangle \\ |1\rangle \quad & \text{as} \quad |111\rangle \end{aligned} \tag{1}$$

This is known as the 3-bit bit-flip quantum error code. This code allows one to detect and correct bit flip errors.

Once we have encoded via equation 1 what does a bit flip error do? We have three qubits. A bit-flip error in the first bit does this

$$\begin{aligned} |000\rangle &\rightarrow |100\rangle \\ |111\rangle &\rightarrow |011\rangle \end{aligned}$$

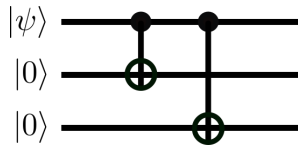

Figure 1: Encoding circuit for the 3-bit bit flip error code of equation 1.

This error is described by a Pauli-X operation on the first qubit or $\mathbf{X} \otimes \mathbf{I} \otimes \mathbf{I}$ where

$$\mathbf{X} = \begin{pmatrix} 0 & 1 \\ 1 & 0 \end{pmatrix}.$$

The Pauli-X operates like this $\mathbf{X}|0\rangle = |1\rangle$ and $\mathbf{X}|1\rangle = |0\rangle$. A bit-flip error in the second bit does this

$$|000\rangle \rightarrow |010\rangle$$
$$|111\rangle \rightarrow |101\rangle$$

This error is described by a Pauli-X operation on the second qubit or $\mathbf{I} \otimes \mathbf{X} \otimes \mathbf{I}$.

A single bit flip error gives a final state that can be detected via a majority rule of 2 out of 3. If there are two zeros and one 1 in the final state vector, then the state should be corrected to $|000\rangle$, whereas if there are two 1s and one 0s then the state should be corrected to $|111\rangle$.

How would we detect this kind of error?

We can perform a measurement that is known as a **syndrome**. A syndrome is a measurement, giving a number that describes the type of error. There are four possible situations for a single bit flipping error:

- There is no error.
- The first bit is flipped.
- The second bit is flipped.
- The third bit is flipped.

To associate a number with these four possibilities we need to perform two measurements each giving a 0 or 1. The outcomes of the measurements are 00, 01, 10 or 11, which is four possible measurements for the possible errors that could have taken place. A circuit that does this is shown in Figure 2.

The transformation performed by the syndrome extraction circuit shown in Figure 2

$$\mathbf{U}_{BF} : |x_0, x_1, x_2, 0, 0\rangle \rightarrow |x_0, x_1, x_2, x_1 + x_2, x_0 + x_2\rangle \tag{2}$$

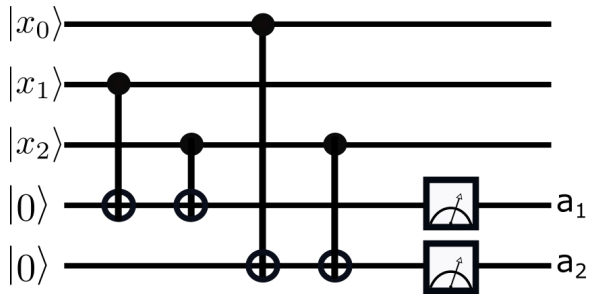

Figure 2: Detecting errors for the 3-bit bit-flip error correction code. The bottom two ancilla qubits are measured to detect a single bit flip error in the top three bits. The circuit is called a syndrome extraction circuit because measurement of the bottom two ancilla qubits gives the syndrome.

Here BF stands for bit-flip. The bottom two bits in the figure are equivalent to the last two bits in the equation and can be called **ancilla** or **ancillary** qubits[1], as they aid in the computation. We refer to the ancilla bits as a_1 and a_2.

We start with initial state $a\,|0\rangle + b\,|1\rangle$ which is encoded as $a\,|000\rangle + b\,|111\rangle$.

Syndrome computation circuit for a Bit Flip Error (Figure 3)				
error	State bits $\|x_0x_1x_2\rangle$	ancilla qubits $\|a_1a_2\rangle$	syndrome a_1a_2	desired correction
None	$a\,\|000\rangle + b\,\|111\rangle$	$\otimes\,\|00\rangle$	00	none
$\mathbf{X}\otimes\mathbf{I}\otimes\mathbf{I}$	$a\,\|100\rangle + b\,\|011\rangle$	$\otimes\,\|01\rangle$	01	flip x_0
$\mathbf{I}\otimes\mathbf{X}\otimes\mathbf{I}$	$a\,\|010\rangle + b\,\|101\rangle$	$\otimes\,\|10\rangle$	10	flip x_1
$\mathbf{I}\otimes\mathbf{I}\otimes\mathbf{X}$	$a\,\|001\rangle + b\,\|110\rangle$	$\otimes\,\|11\rangle$	11	flip x_2

The error detection circuit along with a correction circuit is shown in Figure 3. The correction transformation is

$$\begin{aligned}\mathbf{U}_{BF,corr} = \;& \text{CNOT}(\text{control} = \overline{a}_1 \wedge a_2, \text{target} = x_0) \\ & + \text{CNOT}(\text{control} = a_1 \wedge \overline{a}_2, \text{target} = x_1) \\ & + \text{CNOT}(\text{control} = a_1 \wedge a_2, \text{target} = x_2). \end{aligned} \tag{3}$$

2.1.1 Encoding is not equivalent to cloning

As we showed previously quantum cloning is not possible (via the *no-cloning theorem*). We now show that the encoding procedure of equation 1 is not the same thing as quantum

[1]The word 'ancilla' is conventionally a noun but it seems to be used as an adjective in quantum computing. Sometimes people also use the plural form ancillae. Conventionally the adjective is 'ancillary' but this word seems not much used in the literature on quantum computing.

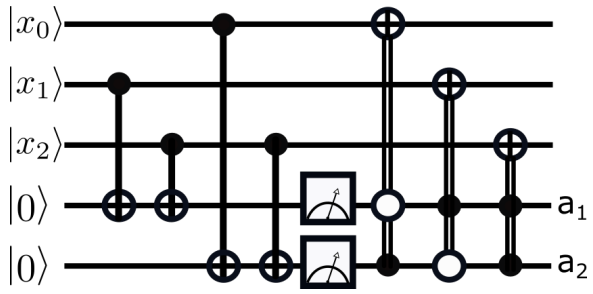

Figure 3: Correcting errors in the 3-bit bit-flip quantum error correction code. The bottom two ancilla qubits are measured to detect a single bit flip error in the top three bits. The three operations on the right correct any of the three possible single bit-flip errors.

cloning.

For the moment we consider only encoding into a two qubit space

$$\begin{aligned} |0\rangle &\to |00\rangle \\ |1\rangle &\to |11\rangle . \end{aligned}$$

Starting with $|\psi\rangle = a\,|0\rangle + b\,|1\rangle$ the encoded state is

$$|\psi\rangle_{\text{encoded}} = a\,|00\rangle + b\,|11\rangle .$$

A cloned state would be

$$\begin{aligned} |\psi\rangle \otimes |\psi\rangle &= (a\,|0\rangle + b\,|1\rangle) \otimes (a\,|0\rangle + b\,|1\rangle) \\ &= a^2\,|00\rangle + b^2\,|11\rangle + ab(|01\rangle + |10\rangle) \\ &\neq a\,|00\rangle + b\,|11\rangle . \end{aligned}$$

We have shown that the encoded state is not the same as the cloned state. By analogy encoding via the 3-big bit-flip error correction code is not the same thing as cloning three times (which is also impossible).

2.1.2 Bit flip error correction on a superposition state

What happens with the full state vector? Suppose the qubit we want to protect initially has

$$|\psi\rangle_1 = a\,|0\rangle + b\,|1\rangle . \tag{4}$$

After coding this single qubit to three qubits, the state is

$$|\psi\rangle_3 = a\,|000\rangle + b\,|111\rangle . \tag{5}$$

Adding in the two ancilla qubits, the initial state is

$$|\psi\rangle_5 = (a\,|000\rangle + b\,|111\rangle) \otimes |00\rangle\,.$$

If the third bit has a bit flip error the state becomes

$$\mathbf{I} \otimes \mathbf{I} \otimes \mathbf{X} \otimes \mathbf{I} \otimes \mathbf{I}\,|\psi\rangle_5 = (a\,|001\rangle + b\,|110\rangle) \otimes |00\rangle\,.$$

Now we run it through the circuit. The second and third bits differ in both substates so the first ancilla bit will become 1. The first and third bits differ in both substates so the second ancilla qubit becomes 1. Prior to measurement the new state is

$$\mathbf{U}_{BF}(\mathbf{I} \otimes \mathbf{I} \otimes \mathbf{X} \otimes \mathbf{I} \otimes \mathbf{I})\,|\psi\rangle_5 = (a\,|001\rangle + b\,|110\rangle) \otimes |11\rangle\,,$$

where $\mathbf{U}_{BF}$ was the error detecting operation defined in equation 2. The measurement gives a syndrome of 11 and the resulting correction performs an $\mathbf{X}$ (or bit flip) on the third bit. After the correction the state is

$$\mathbf{U}_{BF,corr}U_{BF}(\mathbf{I} \otimes \mathbf{I} \otimes \mathbf{X} \otimes \mathbf{I} \otimes \mathbf{I})\,|\psi\rangle_5 = (a\,|000\rangle + b\,|111\rangle) \otimes |11\rangle\,,$$

where the correction operation is given in equation 3. We see that the first 3 bits are restored to the original state in equation 5.

2.1.3 A linear combination of bit flip errors

Suppose there is a linear combination of bit fip errors that occurs. For example suppose the error that occurs is a combination

$$\mathbf{E} = \alpha\ \mathbf{X} \otimes \mathbf{I} \otimes \mathbf{I} + \beta\ \mathbf{I} \otimes \mathbf{X} \otimes \mathbf{I} \tag{6}$$

where α, β are complex numbers that satisfy $\alpha\alpha^* + \beta\beta^* = 1$ so as to ensure that the transformation is unitary. This corresponds to bit flip errors on first and second qubits. Again we start with our single qubit that we want to protect

$$|\psi\rangle_1 = a\,|0\rangle + b\,|1\rangle \tag{7}$$

that is in a general state with a, b complex numbers and normalized. After encoding with three qubits, the initial state is

$$|\psi\rangle_3 = a\,|000\rangle + b\,|111\rangle\,.$$

The error transforms it

$$\mathbf{E}\,|\psi\rangle_3 = \alpha(a\,|100\rangle + b\,|011\rangle) + \beta(a\,|010\rangle + b\,|101\rangle).$$

With 2 ancilla qubits the initial state is

$$\mathbf{E}\,|\psi\rangle_5 = \alpha(a\,|100\rangle + b\,|011\rangle) \otimes |00\rangle + \beta(a\,|010\rangle + b\,|101\rangle) \otimes |00\rangle\,.$$

(Here E includes identity transformations for the ancilla bits). After the error detection circuit is applied (and prior to measurement) the state becomes

$$\mathbf{U}_{BF}\mathbf{E}\,|\psi\rangle_5 = \alpha(a\,|100\rangle + b\,|011\rangle) \otimes |01\rangle + \beta(a\,|010\rangle + b\,|101\rangle) \otimes |10\rangle\,.$$

The measurement can give 01 with probability $\alpha\alpha^*$ or it can give 10 with probability $\beta\beta^*$.

If an 01 is measured then the state vector collapses to $(a\,|100\rangle + b\,|011\rangle) \otimes |01\rangle$.

The correction circuit converts this to $(a\,|000\rangle + b\,|111\rangle) \otimes |01\rangle$.

If an 10 is measured then the state vector collapses to $(a\,|010\rangle + b\,|101\rangle) \otimes |10\rangle$.

The correction circuit converts this to $(a\,|000\rangle + b\,|111\rangle) \otimes |10\rangle$.

In both cases the state vector $|\psi\rangle_3$ in the first 3 bits is restored. The error correction procedure can correct any linear combination of bit flip errors.

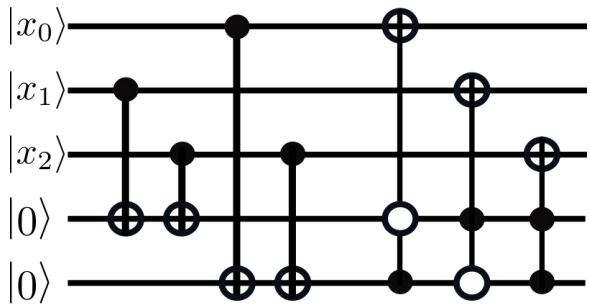

Figure 4: Correcting errors in the 3-bit bit-flip error correction code without measurement. The bottom two ancilla qubits are used to detect a single bit flip error in the top three bits. The three operations on the right correct any of the three possible bit-flip errors. The circuit is the same as in Figure 3 except measurement is neglected. Note that if an error occurs, the ancilla qubits are not in the $|00\rangle$ state at the end of the computation.

2.1.4 Bit flip error correction without measurement

You don't need to actually do the measurement. For example, the circuit in Figure 4 would correct the 3 possible bit flips errors. However, if you want to know if an error occurred, and if so which one occurred, then you would need to measure the two ancilla bits. You could do the measurement after the error correction is completed.

What if the measurement is not performed but instead CNOTs are used to correct the error as shown in the circuit in Figure 4? We check to see what the final state looks like. We use the example in the previous subsection with initial qubit described in equation 7

and with error that is a linear combination that is described in equation 6. The correction circuit gives

$$\mathbf{U}_{BF,corr}\mathbf{U}_{BF}\mathbf{E}\left|\psi\right\rangle_5 = (a\left|000\right\rangle + b\left|111\right\rangle) \otimes (\alpha\left|01\right\rangle + \beta\left|10\right\rangle). \tag{8}$$

Once again the state vector $\left|\psi\right\rangle_3$ in the first 3 bits is restored.

2.1.5 Ancilla qubits must be reset in order to be reused

The error correction scheme requires 3 encoding qubits and 2 ancillary qubits. The 5 qubit space can be considered a tensor product space of the 3 encoding bits and the 2 ancillary bits. The final state after error correction in Equation 8 is not entangled. However the two ancillary bits are not in the $\left|00\right\rangle$ state that they started out with at the beginning of the circuit. If an error occurs and is detected, the ancillary states differ from their initial state.

Question: What happens if we ran the circuit of Figure 4 but did not ensure that the ancillary qubits were initialized in the $\left|00\right\rangle$ state?

Answer: Errors would be falsely detected and the correction circuit would corrupt the encoded state. That means the finally state would not be in the encoded subspace. Ancilla qubits need to be **reset** in order to be reused in an error correction protocol.

2.1.6 Projection operators and measurements for detecting bit-flip errors

The bit flip error measurement can be described with four projection operators.

Error detection or syndrome diagnosis for a Bit Flip Error	
	Projection operator
No error	$\mathbf{P}_0 = \left\|000\right\rangle\left\langle 000\right\| + \left\|111\right\rangle\left\langle 111\right\|$
First bit flipped	$\mathbf{P}_1 = \left\|100\right\rangle\left\langle 100\right\| + \left\|011\right\rangle\left\langle 011\right\|$
Second bit flipped	$\mathbf{P}_2 = \left\|010\right\rangle\left\langle 010\right\| + \left\|101\right\rangle\left\langle 101\right\|$
Third bit flipped	$\mathbf{P}_3 = \left\|001\right\rangle\left\langle 001\right\| + \left\|110\right\rangle\left\langle 110\right\|$

The set of projection operators are Hermitian, orthogonal, and complete (they sum to the identity) so the measurement is a projective measurement. These four projection operators define 4 perpendicular subspaces in the encoded Hilbert space of three qubits.

We describe errors and corrections for them with three single qubit gates; $\mathbf{A} \otimes \mathbf{B} \otimes \mathbf{C}$. We denote the measurement as $\mathbf{A}_1\mathbf{B}_2\mathbf{C}_3$ where the subscript tells you which qubit the single qubit operator operates on. Because these are tensor products, the order is not important. We restrict the possible measurement operators to the Pauli spin matrices and refer to $\mathbf{X} = \sigma_x$, $\mathbf{Y} = \sigma_y$ and $\mathbf{Z} = \sigma_z$. Note we are indexing from 1.

The single bit flip of the first gate would correspond to $\mathbf{X}_1 = \mathbf{X} \otimes \mathbf{I} \otimes \mathbf{I}$. Similarly the error or the correction transformation for a bit flip of the second gate would be $\mathbf{X}_2$.

The projection operators in the above table can be described in terms of two consecutive measurements. One of $\mathbf{Z}_1\mathbf{Z}_2$, the other of $\mathbf{Z}_2\mathbf{Z}_3$. Recall that the Pauli Z matrix

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

The measurements $\mathbf{Z}_1\mathbf{Z}_2$ and $\mathbf{Z}_2\mathbf{Z}_3$ have eigenvalues of ± 1 and each measurement gives 1 bit of of information. So two measurements could give the syndrome we described above. What do these measurement operators look like?

$$\begin{aligned}
\mathbf{Z}_1\mathbf{Z}_2 &= (|0\rangle\langle 0| - |1\rangle\langle 1|) \otimes (|0\rangle\langle 0| - |1\rangle\langle 1|) \otimes \mathbf{I} \\
&= (|00\rangle\langle 00| - |01\rangle\langle 01| - |10\rangle\langle 10| + |11\rangle\langle 11|) \otimes \mathbf{I} \\
&= (|00\rangle\langle 00| + |11\rangle\langle 11|) \otimes \mathbf{I} - (|01\rangle\langle 01| + |10\rangle\langle 10|) \otimes \mathbf{I}
\end{aligned}$$

Notice that the operator gives a 1 if the first two bits agree and gives a -1 if the first two bits do not agree. Similarly $\mathbf{Z}_1\mathbf{Z}_3$ gives a 1 if the first and third bits agree and gives a -1 if the first and third bit do not agree and $\mathbf{Z}_2\mathbf{Z}_3$ gives a 1 if the second and third bits agree and gives a -1 if the second and third bits do not agree.

On the states $|000\rangle$ or $|111\rangle$ the possible measurements are

Syndrome extraction for the bit flip error using operators			
Error	States	Measurement $\mathbf{Z}_1\mathbf{Z}_2$	Measurement $\mathbf{Z}_2\mathbf{Z}_3$
None	$\|000\rangle, \|111\rangle$	1	1
$\mathbf{X}_1$	$\|100\rangle, \|011\rangle$	-1	1
$\mathbf{X}_2$	$\|010\rangle, \|101\rangle$	-1	-1
$\mathbf{X}_3$	$\|001\rangle, \|110\rangle$	1	-1

If you add 1 and divide by two for each measurement (giving $1 \to 1, -1 \to 0$), the series of measurements becomes 11,01,00,10. Notice that the measurement values do not depend upon the values a, b describing the superposition state of the state vector $|\psi\rangle_3 = a|000\rangle + b|111\rangle$. We can create a syndrome for bit flip error correction that is based on measurements of two pairs of non-identical $\mathbf{Z}$ operators.

2.1.7 Phase errors are not detected by the 3-bit bit-flip error correction code

Instead of an error described by a Pauli X operation, suppose instead it is described by the Pauli Z operator which changes the sign of a $|1\rangle$ state. Note that $-1 = e^{\pi i}$ and can be thought of a phase. As before, we encode $|\psi\rangle = a|0\rangle + b|1\rangle$ as $|\psi\rangle_3 = a|000\rangle + b|111\rangle$. A phase error in the first bit sends

$$|000\rangle \to |000\rangle \qquad |111\rangle \to -|111\rangle$$

The error is described with $\mathbf{Z}$ on the first bit so

$$\mathbf{Z} \otimes \mathbf{I} \otimes \mathbf{I} \left|\psi\right\rangle_3 = a \left|000\right\rangle - b \left|111\right\rangle .$$

However our error detection circuit in Figure 2 that finds bit flip errors will fail to detect this phase error. The error correction circuit shown in Figure 3 would return $a \left|000\right\rangle - b \left|111\right\rangle$ in the top three bits.

2.1.8 Subspaces

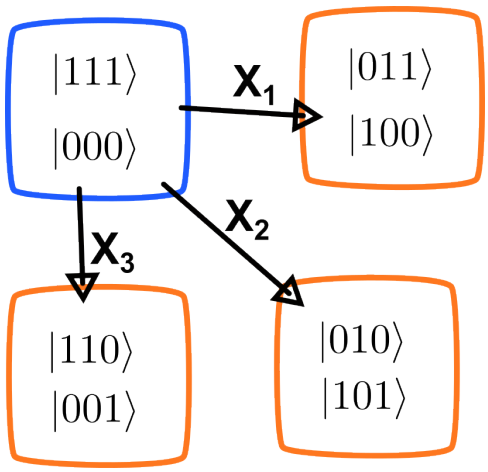

Figure 5: Each rectangle shows a two dimensional vector subspace that is spanned by basis vectors that are within the rectangle. The subspaces are perpendicular to each other. The blue rectangle shows the encoding subspace for the bit-flip error correction code. The different bit flip errors are shown with black arrows. Bit-flip errors take a state in the encoding subspace to a state within a perpendicular subspace. The syndrome measures which subspace a state is found in, and does not measure information about the actual state within that subspace. A state can be restored to the encoding subspace without affecting the phase information within the subspace.

The following identities

$$\mathbf{HXH} = \mathbf{Z} \tag{9}$$

$$\mathbf{HZH} = \mathbf{X}. \tag{10}$$

help us understand the similarity between the three bit bit-flip and the three bit phase-flip error correcting codes.

2.2 Correcting phase flip errors with a three qubit code

We discuss a three-bit error correction code designed to correct for phase flip errors.

In this case we encode

$$\begin{aligned} |0\rangle &\to |+++\rangle \\ |1\rangle &\to |---\rangle . \end{aligned} \tag{11}$$

Here $|+\rangle = \frac{1}{\sqrt{2}}(|0\rangle + |1\rangle)$ and $|-\rangle = \frac{1}{\sqrt{2}}(|0\rangle - |1\rangle)$.

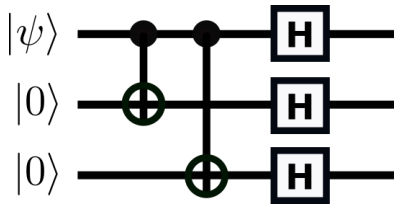

Figure 6: Encoding circuit for the 3-bit phase flip error code of equation 11.

A sign (or phase) error in a single bit is described by the Pauli-Z gate. $\mathbf{Z}|0\rangle = |0\rangle$ and $\mathbf{Z}|1\rangle = -|1\rangle$.

$$\begin{aligned} \mathbf{Z}|+\rangle &= Z\frac{1}{\sqrt{2}}(|0\rangle + |1\rangle) = \frac{1}{\sqrt{2}}(|0\rangle - |1\rangle) \\ &= |-\rangle \\ \mathbf{Z}|-\rangle &= Z\frac{1}{\sqrt{2}}(|0\rangle - |1\rangle) = \frac{1}{\sqrt{2}}(|0\rangle + |1\rangle) \\ &= |+\rangle \end{aligned}$$

The Pauli $\mathbf{Z}$ gate swaps the $|+\rangle, |-\rangle$ states. The Hadamard gate $\mathbf{H}$ transfers from the $\{|0\rangle, |1\rangle\}$ basis to the $\{|+\rangle, |-\rangle\}$ basis. That is why, with the addition of Hadamard gates, Figure 6 resembles Figure 1 the encoding circuit for the 3-bit bit-flip error correcting code.

Performing phase errors on single bits

$$\begin{aligned} \mathbf{Z}_1|+++\rangle &= |-++\rangle & \mathbf{Z}_1|---\rangle &= |+--\rangle \\ \mathbf{Z}_2|+++\rangle &= |+-+\rangle & \mathbf{Z}_2|---\rangle &= |-+-\rangle \\ \mathbf{Z}_3|+++\rangle &= |++-\rangle & \mathbf{Z}_3|---\rangle &= |--+\rangle . \end{aligned}$$

Syndrome measurement can be done using these two operators

$$\begin{aligned} \mathbf{H}^{\otimes 3}\mathbf{Z}_1\mathbf{Z}_2\mathbf{H}^{\otimes 3} &= \mathbf{X}_1\mathbf{X}_2 \\ \mathbf{H}^{\otimes 3}\mathbf{Z}_2\mathbf{Z}_3\mathbf{H}^{\otimes 3} &= \mathbf{X}_2\mathbf{X}_3 \end{aligned}$$

where $\mathbf{H}^{\otimes 3} = \mathbf{H} \otimes \mathbf{H} \otimes \mathbf{H}$.

Let's check that this makes sense by looking at the two encoding state.

$$|+++\rangle = \frac{1}{\sqrt{8}}(|000\rangle + |001\rangle + |010\rangle + |011\rangle + |100\rangle + |101\rangle + |110\rangle + |111\rangle)$$
$$|---\rangle = \frac{1}{\sqrt{8}}(|000\rangle - |001\rangle - |010\rangle + |011\rangle - |100\rangle + |101\rangle + |110\rangle - |111\rangle)$$

We notice that the sign depends on the parity of the state. With no phase flip error, if we flip both the first and second bit then both states are unchanged.

We now exert a phase error in the first bit.

$$|-++\rangle = \frac{1}{\sqrt{8}}(|000\rangle + |001\rangle + |010\rangle + |011\rangle - |100\rangle - |101\rangle - |110\rangle - |111\rangle)$$
$$|+--\rangle = \frac{1}{\sqrt{8}}(|000\rangle - |001\rangle - |010\rangle + |011\rangle + |100\rangle - |101\rangle - |110\rangle + |111\rangle)$$

Flipping the first and second bits with $\mathbf{X}_1\mathbf{X}_2$ causes both states to change sign. However, flipping the second and third bits with $\mathbf{X}_2\mathbf{X}_3$ will not affect the states. This gives measurement (in terms of eigenvalues) of -1, 1. Using the syndrome measurement, a single bit phase flip error can be identified and corrected.

Syndrome extraction for the phase flip error using operators			
Error	States	Measurement $\mathbf{X}_1\mathbf{X}_2$	Measurement $\mathbf{X}_2\mathbf{X}_3$
None	$\|+++\rangle, \|---\rangle$	1	1
$\mathbf{Z}_1$	$\|-++\rangle, \|+--\rangle$	-1	1
$\mathbf{Z}_2$	$\|-+-\rangle, \|-+-\rangle$	-1	-1
$\mathbf{Z}_3$	$\|++-\rangle, \|++-\rangle$	1	-1

3 Shor's 9-bit code

In section 2.1 we illustrated a 3-bit code that can correct bit-flip errors but not phase flip errors. In section 2.2 we illustrated a 3-bit code that can correct phase flip errors, but it cannot correct bit-flip errors. The Shor 9-bit code uses 9 bits to correct both types of errors.

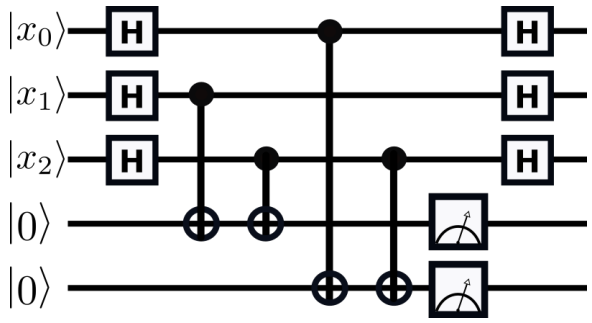

Figure 7: Detecting errors in the 3-bit phase-flip error correction code. The bottom two ancilla qubits are measured to detect a single **phase** flip error in the top three bits. The circuit is similar to Figure 2 for the bit flip syndrome extraction circuit.

3.1 Shor's 9-bit code

The proposal is to encode

$$\begin{aligned} |0\rangle \quad &\text{as} \quad \frac{1}{2^{3/2}}(|000\rangle + |111\rangle) \otimes (|000\rangle + |111\rangle) \otimes (|000\rangle + |111\rangle) \\ |1\rangle \quad &\text{as} \quad \frac{1}{2^{3/2}}(|000\rangle - |111\rangle) \otimes (|000\rangle - |111\rangle) \otimes (|000\rangle - |111\rangle). \end{aligned} \tag{12}$$

There are 9 states. You can think of the 9 states as a 3×3 matrix where each row protects against bit flip errors and each column protects agains phase flip errors. In other words

$$|0\rangle \rightarrow \frac{1}{2^{3/2}}\left(\left|\begin{matrix}000\\000\\000\end{matrix}\right\rangle + \left|\begin{matrix}000\\000\\111\end{matrix}\right\rangle + \left|\begin{matrix}000\\111\\000\end{matrix}\right\rangle + \left|\begin{matrix}000\\111\\111\end{matrix}\right\rangle + \left|\begin{matrix}111\\000\\000\end{matrix}\right\rangle + \left|\begin{matrix}111\\000\\111\end{matrix}\right\rangle + \left|\begin{matrix}111\\111\\000\end{matrix}\right\rangle + \left|\begin{matrix}111\\111\\111\end{matrix}\right\rangle\right)$$

$$|1\rangle \rightarrow \frac{1}{2^{3/2}}\left(\left|\begin{matrix}000\\000\\000\end{matrix}\right\rangle - \left|\begin{matrix}000\\000\\111\end{matrix}\right\rangle - \left|\begin{matrix}000\\111\\000\end{matrix}\right\rangle + \left|\begin{matrix}000\\111\\111\end{matrix}\right\rangle - \left|\begin{matrix}111\\000\\000\end{matrix}\right\rangle + \left|\begin{matrix}111\\000\\111\end{matrix}\right\rangle + \left|\begin{matrix}111\\111\\000\end{matrix}\right\rangle - \left|\begin{matrix}111\\111\\111\end{matrix}\right\rangle\right)$$

We discuss protecting agains bit flip errors. We first consider the first three bits. We apply measurements $\mathbf{Z}_1\mathbf{Z}_2$ and $\mathbf{Z}_2\mathbf{Z}_3$. These two observables have eigenvalues of 1 and -1. Recall that if $|\psi\rangle$ is an eigenvector of $\mathbf{Z}_1\mathbf{Z}_2$, then its eigenvalue is $\langle\psi|\,\mathbf{Z_1Z_2}\,|\psi\rangle$. The eigenvalues of an operator are the quantities that are measured by that operator.

If there is a bit flip error in the first bit we would notice that measurement of $\mathbf{Z}_1\mathbf{Z}_2$ gives an eigenvalue of -1 but measurement of $\mathbf{Z}_2\mathbf{Z}_3$ gives eigenvalue 1. This is similar to the bit flip measurements discussed in section 2.1.6. Measurements of

$$\mathbf{Z}_1\mathbf{Z}_2 \qquad \text{and} \qquad \mathbf{Z}_2\mathbf{Z}_3$$

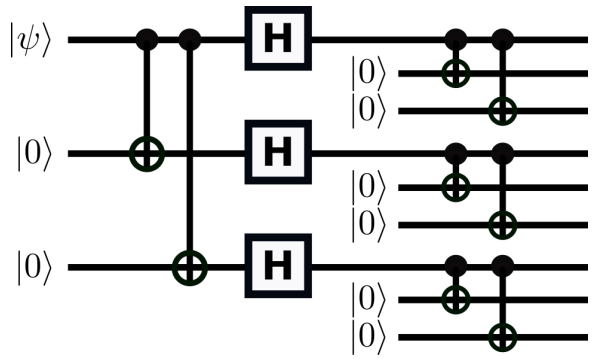

Figure 8: Encoding circuit for Shor's 9 qubit error code.

can be used to create a 2 bit syndrome that would find a bit flip error in the first set of three bits. Similarly

$$\mathbf{Z}_4\mathbf{Z}_5 \qquad \text{and} \qquad \mathbf{Z}_5\mathbf{Z}_6$$

can be used to create a 2 bit syndrome that would find a bit flip error in the second set of three bits. Similarly

$$\mathbf{Z}_7\mathbf{Z}_8 \qquad \text{and} \qquad \mathbf{Z}_8\mathbf{Z}_9$$

can be used to create a 2 bit syndrome that would find a bit flip error in the third set of three bits.

Using the eigenvalues for the operators,

Syndrome for bit flip errors in the 9-bit code							
Error	Measurements						Correction
	$\mathbf{Z}_1\mathbf{Z}_2$	$\mathbf{Z}_2\mathbf{Z}_3$	$\mathbf{Z}_4\mathbf{Z}_5$	$\mathbf{Z}_5\mathbf{Z}_6$	$\mathbf{Z}_7\mathbf{Z}_8$	$\mathbf{Z}_8\mathbf{Z}_9$	
None	1	1	1	1	1	1	None
$\mathbf{X}_1$	-1	1	1	1	1	1	$\mathbf{X}_1$
$\mathbf{X}_2$	-1	-1	1	1	1	1	$\mathbf{X}_2$
$\mathbf{X}_3$	1	-1	1	1	1	1	$\mathbf{X}_3$
$\mathbf{X}_4$	1	1	-1	1	1	1	$\mathbf{X}_4$
$\mathbf{X}_5$	1	1	-1	-1	1	1	$\mathbf{X}_5$
$\mathbf{X}_6$	1	1	1	-1	1	1	$\mathbf{X}_6$
$\mathbf{X}_7$	1	1	1	1	-1	1	$\mathbf{X}_7$
$\mathbf{X}_8$	1	1	1	1	-1	-1	$\mathbf{X}_8$
$\mathbf{X}_9$	1	1	1	1	1	-1	$\mathbf{X}_9$

What happens if there is a phase flip error? The phase flip error involves the clusters of 3 qubits. The observables

$$\mathbf{X}_1\mathbf{X}_2\mathbf{X}_3\mathbf{X}_4\mathbf{X}_5\mathbf{X}_6 \qquad \text{and} \qquad \mathbf{X}_4\mathbf{X}_5\mathbf{X}_6\mathbf{X}_7\mathbf{X}_8\mathbf{X}_9 \tag{13}$$

also have eigenvalues ± 1.

For example consider an phase flip error in the first qubit with an initial state $|\psi\rangle = a|0\rangle + b|1\rangle$ that is encoded via the Shor 9 bit code to $|\psi\rangle_9$,

$$\begin{aligned}\mathbf{Z}_1|\psi\rangle_9 =& \mathbf{Z}_1[a(|000\rangle + |111\rangle)^{\otimes 3} + b(|000\rangle - |111\rangle)^{\otimes 3}] \\ =& a(|000\rangle - |111\rangle) \otimes (|000\rangle + |111\rangle)^{\otimes 2} \\ &+ b(|000\rangle + |111\rangle) \otimes (|000\rangle - |111\rangle)^{\otimes 2}.\end{aligned}$$

If we operate on this with $\mathbf{X}_1\mathbf{X}_2\mathbf{X}_3\mathbf{X}_4\mathbf{X}_5\mathbf{X}_6$ the result is

$$\begin{aligned}\mathbf{X}_1\mathbf{X}_2\mathbf{X}_3\mathbf{X}_4\mathbf{X}_5\mathbf{X}_6\mathbf{Z}_1|\psi\rangle_9 =& a(|111\rangle - |000\rangle) \otimes (|000\rangle + |111\rangle)^{\otimes 2} \\ &+ b(|000\rangle + |111\rangle) \otimes (|111\rangle - |000\rangle) \otimes (|000\rangle - |111\rangle \\ =& -a(|000\rangle - |111\rangle) \otimes (|000\rangle + |111\rangle)^{\otimes 2} \\ &- b(|000\rangle + |111\rangle) \otimes (|000\rangle - |111\rangle)^{\otimes 2} \\ =& -\mathbf{Z}_1|\psi\rangle_9 .\end{aligned}$$

The state $\mathbf{Z}_1|\psi\rangle_9$ is an eigenvector of $\mathbf{X}_1\mathbf{X}_2\mathbf{X}_3\mathbf{X}_4\mathbf{X}_5\mathbf{X}_6$ with eigenvalue -1. A measurement of $\mathbf{X}_1\mathbf{X}_2\mathbf{X}_3\mathbf{X}_4\mathbf{X}_5\mathbf{X}_6$ on the state where a phase flip error ocured on the first bit would give -1.

The two observables in equation 13 create a 2 bit syndrome that would find a phase flip error in the 3 clusters of 3 qubits. Because only two bits are used to detect the error, the corrections must be able to fix a phase error in any of the three bits in a cluster. But we notice that

$$\mathbf{Z}_1(|000\rangle + |111\rangle) = \mathbf{Z}_2(|000\rangle + |111\rangle) = \mathbf{Z}_3(|000\rangle + |111\rangle).$$

Similarly

$$\mathbf{Z}_1(|000\rangle - |111\rangle) = \mathbf{Z}_2(|000\rangle - |111\rangle) = \mathbf{Z}_3(|000\rangle - |111\rangle).$$

So an operation of $\mathbf{Z}_1\mathbf{Z}_2\mathbf{Z}_3$ can be used to correct a phase flip error in the cluster of the first three bits.

Syndrome for phase flip errors in the 9-bit code			
Error	Measurements		Correction
	$\mathbf{X}_1\mathbf{X}_2\mathbf{X}_3\mathbf{X}_4\mathbf{X}_5\mathbf{X}_6$	$\mathbf{X}_4\mathbf{X}_5\mathbf{X}_6\mathbf{X}_7\mathbf{X}_8\mathbf{X}_9$	
None	1	1	None
$\mathbf{Z}_1$ or $\mathbf{Z}_2$ or $\mathbf{Z}_3$	-1	1	$\mathbf{Z}_1\mathbf{Z}_2\mathbf{Z}_3$
$\mathbf{Z}_4$ or $\mathbf{Z}_5$ or $\mathbf{Z}_6$	-1	-1	$\mathbf{Z}_4\mathbf{Z}_5\mathbf{Z}_6$
$\mathbf{Z}_7$ or $\mathbf{Z}_8$ or $\mathbf{Z}_9$	1	-1	$\mathbf{Z}_7\mathbf{Z}_8\mathbf{Z}_9$

3.2 Correcting all 1-qubit errors

If a quantum error-correcting code protects against both bit-flip and sign/phase errors, then the code automatically protects against all possible 1-qubit errors.

We show that this is true for Shor's 9-bit code that encodes

$$|0\rangle \rightarrow (|000\rangle + |111\rangle)^{\otimes 3} = \left|\tilde{0}\right\rangle$$
$$|1\rangle \rightarrow (|000\rangle - |111\rangle)^{\otimes 3} = \left|\tilde{1}\right\rangle$$

The encoded state-vector has 9 bits. However, the initial encoded state lives in a subspace of the full 2^9 dimensional Hilbert space for the 9-bit code. We call the subspace that is spanned by the initial encoded state C. It is spanned by the two state vectors $\left|\tilde{0}\right\rangle = (|000\rangle + |111\rangle)^{\otimes 3}$ and $\left|\tilde{1}\right\rangle = (|000\rangle - |111\rangle)^{\otimes 3}$, as any vector within the subspace C can be written as a sum of these two vectors.

We describe errors in the form $\mathbf{E}_i$ where $i \in \{1, 2, 3, 4, ...\}$ and each error is a single qubit operation that is a Pauli matrix.

A more general error on a single qubit can be written as a unitary transformation Q acting on a single bit. Recall that any unitary transformation can be written in the following form using 4 angles

$$\begin{aligned}\mathbf{Q} &= e^{i\delta}\begin{pmatrix} \cos\beta\ e^{i(\alpha+\gamma)} & \sin\beta\ e^{i(\alpha-\gamma)} \\ -\sin\beta\ e^{-i(\alpha-\gamma)} & \cos\beta\ e^{-i(\alpha+\gamma)}\end{pmatrix} \\ &= e^{i\delta}\Big[\cos\beta\cos(\alpha+\gamma)\mathbf{I} + i\cos\beta\sin(\alpha+\gamma)\mathbf{Z} \\ &\qquad + i\sin\beta\sin(\alpha-\gamma)\mathbf{X} + i\sin\beta\cos(\alpha-\gamma)\mathbf{Y}\Big].\end{aligned}$$

Note we can ignore the global phase δ. If a code corrects phase flip and bit-flip errors then it corrects the second and third of these terms. We need only show that an error caused by $\mathbf{Y}$ on any qubit would be corrected by the code.

Suppose a $i\mathbf{Y}_1$ error occurs. It is convenient to use $i\mathbf{Y} = \begin{pmatrix} 0 & 1 \\ -1 & 0\end{pmatrix}$, giving $i\mathbf{Y}\,|0\rangle = -\,|1\rangle$ and $i\mathbf{Y}\,|1\rangle = |0\rangle$. Also $i\mathbf{Y} = \mathbf{ZX}$ so it can be applied by first applying a NOT (equivalent to an $\mathbf{X}$ which bit flips) and then applying $\mathbf{Z}$.

$$i\mathbf{Y}_1\left|\tilde{0}\right\rangle = \frac{1}{\sqrt{8}}(-\,|100\rangle + |011\rangle) \otimes (|000\rangle + |111\rangle) \otimes (|000\rangle + |111\rangle)$$
$$i\mathbf{Y}_1\left|\tilde{1}\right\rangle = \frac{1}{\sqrt{8}}(-\,|100\rangle - |011\rangle) \otimes (|000\rangle - |111\rangle) \otimes (|000\rangle - |111\rangle).$$

We take initial state be $|\psi\rangle = a\left|\tilde{0}\right\rangle + b\left|\tilde{1}\right\rangle$. The state with error applied

$$\begin{aligned}i\mathbf{Y}_1\,|\psi\rangle = a\frac{1}{\sqrt{8}}&(-\,|100\rangle + |011\rangle) \otimes (|000\rangle + |111\rangle) \otimes (|000\rangle + |111\rangle) \\ + b\frac{1}{\sqrt{8}}&(-\,|100\rangle - |011\rangle) \otimes (|000\rangle - |111\rangle) \otimes (|000\rangle - |111\rangle).\end{aligned}$$

The $\mathbf{Z}_1\mathbf{Z}_2$ syndrome will detect an error in both terms. We can correct the error by applying $\mathbf{X}_1$. This will give

$$\begin{aligned}\mathbf{X}_1 i\mathbf{Y}_1 |\psi\rangle = a\frac{1}{\sqrt{8}}(-|000\rangle + |111\rangle) \otimes (|000\rangle + |111\rangle) \otimes (|000\rangle + |111\rangle) \\ + b\frac{1}{\sqrt{8}}(-|000\rangle - |111\rangle) \otimes (|000\rangle - |111\rangle) \otimes (|000\rangle - |111\rangle).\end{aligned}$$

The state is still corrupted.

However, now the $\mathbf{X}_1\mathbf{X}_2\mathbf{X}_3\mathbf{X}_4\mathbf{X}_5\mathbf{X}_6$ syndrome will detect a phase error. We can correct it by applying $\mathbf{Z}_1$ (or by applying $\mathbf{Z}_1\mathbf{Z}_2\mathbf{Z}_3$) giving

$$\begin{aligned}\mathbf{Z}_1\mathbf{X}_1 i\mathbf{Y}_1 |\psi\rangle &= -a\frac{1}{\sqrt{8}}(|000\rangle |111\rangle) \otimes (|000\rangle + |111\rangle) \otimes (|000\rangle + |111\rangle) \\ &\quad - b\frac{1}{\sqrt{8}}(|000\rangle - |111\rangle) \otimes (|000\rangle - |111\rangle) \otimes (|000\rangle - |111\rangle) \\ &= -|\psi\rangle .\end{aligned}$$

Up to a global phase, the error $i\mathbf{Y}_1$ error has been corrected.

We can simultaneously detect both $\mathbf{X}$ and $\mathbf{Z}$ errors. In other words if both $\mathbf{X}_1$ and $\mathbf{Z}_1$ errors occur simultaneously, we can tell from the syndrome bits. As $\mathbf{XZ} = -i\mathbf{Y}$ this means that the error correction code would also correct any $\mathbf{Y}$ error. Any code that corrects all bit-flip errors $\mathbf{X}$ and all phase-flip errors $\mathbf{Z}$ also corrects all $\mathbf{Y}$ errors.

Because any unitary transformation on a single qubit can be written as a linear combination of $\mathbf{I}, \mathbf{X}, \mathbf{Y}, \mathbf{Z}$ operators, a code that corrects all $\mathbf{X}, \mathbf{Y}, \mathbf{Z}$ errors also corrects for any possible unitary transformation on any single qubit.

A nice feature of quantum error-correction is that by correcting just a discrete set of errors, the bit flip, phase flip, and combined bit-phase flip, a quantum error-correcting code is able to automatically correct a much larger and continuous class of errors.

3.3 Error Generated Subspaces

What is the effect of an error on the encoded vector space C? With error $\mathbf{E}_i$ described by a Pauli matrix on one the qubits given by index i, what vector subspace does a state-vector $\mathbf{E}_i |v\rangle$ live in, if $|v\rangle \in C$?

In the Shor 9 bit code, the subspaces $\mathbf{E}_i |v\rangle$ (with $|v\rangle \in C$) are perpendicular to C for all single qubit errors $\mathbf{E}_i$. If this were not true, then we would not be able to tell the difference between a state that is caused by an error and a state that has not been affected by error. This means that for any $|v\rangle, |v'\rangle \in C$ and any single qubit error $\mathbf{E}_i$,

$$\langle v' | \mathbf{E}_i |v\rangle = 0. \tag{14}$$

This is true for the Shor 9-bit code. Consider errors $\mathbf{Z}_1$ or $\mathbf{Z}_2$ or $\mathbf{Z}_3$. One of the three clusters of 3 bits would flip in sign. One of the first 3 bits suffers a phase flip error

$$(|000\rangle + |111\rangle)^{\otimes 3} \rightarrow (|000\rangle - |111\rangle) \otimes (|000\rangle + |111\rangle) \otimes (|000\rangle + |111\rangle)$$

This state is perpendicular to the initial state, and it is also perpendicular to $(|000\rangle - |111\rangle)^{\otimes 3}$. Likewise $\mathbf{Z}_i(|000\rangle - |111\rangle)^{\otimes 3}$ gives a state that is perpendicular to both $(|000\rangle + |111\rangle)^{\otimes 3}$ and $(|000\rangle - |111\rangle)^{\otimes 3}$. The operators $\mathbf{X}_i$ and $\mathbf{Y}_i$ flip bits and when operating on vectors in C they similarly generate subspaces that are perpendicular or orthogonal to C.

Furthermore, pairs of subspaces generated by single qubit errors, when operating on C, are either the same or perpendicular to each other. For the Shor 9 bit code, each of the X_i matrices generates a unique 2 dimensional subspace when operating on C. These subspaces are perpendicular to each other and perpendicular to C. In other words for all $|v\rangle, |v'\rangle \in C$

$$\langle v' | \mathbf{X}_i^\dagger \mathbf{X}_j |v\rangle = 0 \qquad \text{for} \quad i \neq j.$$

The subspaces generated by the set of Pauli Z_i matrices are not all perpendicular to each other as the three $\mathbf{Z}_1, \mathbf{Z}_2, \mathbf{Z}_3$ act similarly on the encoding subspace C. Each of them gives the same subspace. Likewise $\mathbf{Z}_4, \mathbf{Z}_5, \mathbf{Z}_6$ and $\mathbf{Z}_7, \mathbf{Z}_8, \mathbf{Z}_9$ each give another subspace. The subspaces generated by these three groups are perpendicular.

Note that $\mathbf{Z}_1 |v\rangle = \mathbf{Z}_2 |v\rangle = \mathbf{Z}_3 |v\rangle$ for all $|v\rangle \in C$. So if an error is detected, any of the operators $\mathbf{Z}_1, \mathbf{Z}_2$ or $\mathbf{Z}_3$ could be used to correct it. For the Shor 9-bit code

$$\begin{aligned}
\mathbf{Z}_1 |\tilde{0}\rangle &= \frac{1}{\sqrt{8}}(|000\rangle - |111\rangle) \otimes (|000\rangle + |111\rangle) \otimes (|000\rangle + |111\rangle) \\
\mathbf{Z}_2 |\tilde{0}\rangle &= \frac{1}{\sqrt{8}}(|000\rangle - |111\rangle) \otimes (|000\rangle + |111\rangle) \otimes (|000\rangle + |111\rangle) \\
\mathbf{Z}_1 |\tilde{1}\rangle &= \frac{1}{\sqrt{8}}(|000\rangle + |111\rangle) \otimes (|000\rangle - |111\rangle) \otimes (|000\rangle - |111\rangle) \\
\mathbf{Z}_2 |\tilde{1}\rangle &= \frac{1}{\sqrt{8}}(|000\rangle + |111\rangle) \otimes (|000\rangle - |111\rangle) \otimes (|000\rangle - |111\rangle)
\end{aligned}$$

We notice that $\mathbf{Z}_1 |\tilde{0}\rangle = \mathbf{Z}_2 |\tilde{0}\rangle$ and $\mathbf{Z}_1 |\tilde{1}\rangle = \mathbf{Z}_2 |\tilde{1}\rangle$. Furthermore $\mathbf{Z}_1 |\tilde{0}\rangle$ is perpendicular to $\mathbf{Z}_1 |\tilde{1}\rangle$. In other words

$$\langle v_a | \mathbf{Z}_1 \mathbf{Z}_2 |v_b\rangle = \delta_{ab}$$

where $|v_a\rangle, |v_b\rangle$ are basis vectors. This implies that an error caused by $\mathbf{Z}_1$ can be corrected by applying $\mathbf{Z}_2$ without corrupting the encoded state.

3.4 Correctable sets of errors

We take $\{|v_a\rangle\}$ (for different a) as an orthonormal basis for the encoded subspace C. We consider a set of errors $S_E = \{\mathbf{E}_1, \mathbf{E}_2 ....\}$ where each error is a unitary transformation. A

set of errors $S_E = \{\mathbf{E}_1, \mathbf{E}_2....\}$ for a quantum code C is considered **correctible** if there exists numbers m_{ij} such that

$$\langle v_a | \mathbf{E}_i^\dagger \mathbf{E}_j | v_b \rangle = m_{ij} \delta_{ab} \tag{15}$$

for all basis vectors $|v_a\rangle, |v_b\rangle$ in the orthonormal basis for C and all $E_i \in S_E$. Here the m_{ij} are numbers that do not depend upon the basis vectors but do depend upon the error operators.

What happens if this condition is violated? Suppose

$$\mathbf{E}_j | v_b \rangle = \mathbf{E}_i | v_a \rangle \qquad \text{for} \qquad a \neq b.$$

This means that the error E_j and E_i send states in C to the same subspace, but the basis vectors in C are sent to different states in that subspace. Multiply both sides by $E_i^\dagger$ giving

$$\mathbf{E}_i^\dagger \mathbf{E}_j | v_b \rangle = | v_a \rangle .$$

$\mathbf{E}_i^\dagger \mathbf{E}_j$ can be considered a **correction** done by the operator $\mathbf{E}_i$ of an **error** that was caused by $\mathbf{E}_j$. If $a \neq b$ and $i \neq j$, the two errors cause a change of the basis state in the encoding subspace C. If we tried to correct the error $\mathbf{E}_j$ with the operation given by $\mathbf{E}_i$ we would mess up the encoded vector within the encoding space C.

The condition in equation 15 ensures that when two or more errors send the encoded space C to the same subspace, any of the errors can be corrected by applying a single error correction transformation to the encoded state-vector.

The condition in equation 15 can also be discussed in terms of information. Measuring the syndrome should not provide any information about the encoded quantum state, otherwise it would perform a measurement on it and superposition would be destroyed. This implies that

$$\langle v_a | \mathbf{E}_i^\dagger \mathbf{E}_j | v_a \rangle = \langle v_b | \mathbf{E}_i^\dagger \mathbf{E}_j | v_b \rangle$$

for every pair $|v_a\rangle, |v_b\rangle$ of basis vectors in the encoding suspace C and every pair $\mathbf{E}_i, \mathbf{E}_j$ of errors in the error set. (**Notice!!** a is on the left and b is on the right). The values resulting from set of pairs are the coefficients m_{ij} in equation 15.

4 Theory of Quantum Error-Correction

A quantum error correcting code performs the following operations.

- A transformation that encodes the initial state.
- A series of error syndromes are performed to diagnose errors.
- A series of recovery operations that are performed to correct the errors.

4.1 Encoding

The encoding is done via a unitary transformation, however because most of the initial state is determined by initially setting additional bits to $|0\rangle$, as in Figure 12, and performing a unitary transformation, the resulting state resides in a subspace C of the full Hilbert space.

For example the encoding transformation in equation 12 for Shor's 9-bit code can be described in terms of a unitary transformation that we can read off of the encoding circuit in Figure 8. Indexing from 1 and with first bit on the top of the figure, the transformation is

$$U_{9,encode} = \mathrm{CNOT}_{c=1}^{t=2}\mathrm{CNOT}_{c=1}^{t=3}\mathrm{CNOT}_{c=4}^{t=5}\mathrm{CNOT}_{c=7}^{t=8}\mathrm{CNOT}_{c=7}^{t=9}H_1H_4H_7\mathrm{CNOT}_{c=1}^{t=7}\mathrm{CNOT}_{c=1}^{t=4}.$$

Here $c =$ refers to the control bit and $t =$ refers to the target bit. The encoded state is $U_{9,encode}\,|\psi\rangle \otimes |00000000\rangle$ where $|\psi\rangle$ is the single qubit that is encoded. Because the encoding unitary transformation is performed on a state that has 8 bits initialized to 0, the resulting encoded state is restricted to a 2 dimensional subspace of the full 9 bit encoding Hilbert space.

The restriction to a subspace can be described in terms of a projection operator. For example, for the three bit, bit-flip protecting code described in section 2.1, encoding is done with

$$|0\rangle \to |000\rangle \qquad \text{and} \qquad |1\rangle \to |111\rangle\,.$$

After encoding the state $a\,|0\rangle + b\,|1\rangle$ looks like $|\psi\rangle_3 = a\,|000\rangle + b\,|111\rangle$. A projection operator that describes the subspace in which $|\psi\rangle_3$ initially resides is

$$\mathbf{P}_{BF,encode} = |000\rangle\langle 000| + |111\rangle\langle 111|\,.$$

The 3 bit phase correction code described in section 2.2 encoding is done with

$$|0\rangle \to |+++\rangle \qquad \text{and} \qquad |1\rangle \to |---\rangle\,.$$

After encoding the state looks like $|\psi\rangle_3 = a\,|+++\rangle + b\,|---\rangle$. A projection operator that describes the subspace in which $|\psi\rangle_3$ resides is

$$\mathbf{P}_{PF,encode} = |+++\rangle\langle +++| + |---\rangle\langle ---|\,.$$

For the Shor 9-bit code encoding is done via

$$|0\rangle \quad \text{as} \quad |\psi_a\rangle \tag{16}$$

$$|1\rangle \quad \text{as} \quad |\psi_b\rangle \tag{17}$$

where

$$|\psi_a\rangle = \frac{1}{2^{3/2}}(|000\rangle + |111\rangle) \otimes (|000\rangle + |111\rangle) \otimes (|000\rangle + |111\rangle)$$

$$|\psi_b\rangle = \frac{1}{2^{3/2}}(|000\rangle - |111\rangle) \otimes (|000\rangle - |111\rangle) \otimes (|000\rangle - |111\rangle).$$

A projection operator that describes the subspace in which $|\psi\rangle_9$ resides is

$$\mathbf{P}_{S9,encode} = |\psi_a\rangle \langle\psi_a| + |\psi_b\rangle \langle\psi_b| .$$

We have described the space C in which the encoded state vectors reside with projection operators. These specify subspaces of the encoded space.

4.2 Double bracket notation to describe codes

We use the notation $[[n, k]]$ when n qubits are used to encode k qubits of initial data. The initial data resides in a 2^k dimensional Hilbert space. The coded data resides in a 2^n dimensional Hilbert space. For example, the 3 bit bit-flip correction code is [[3,1]] and the 3 bit phase flip correction code is also [[3,1]]. Shor's 9 bit code is [[9,1]]. Similar notation, but only involving a single bracket set, is used for classical error correction codes.

The code block C (which we called the encoding space) has dimension 2^k. This is the vector space in which the initial encoded state resides within the full 2^n dimensional Hilbert space for coded data. A correctible error takes a vector that resides within the encoding space C to another vector that is outside C. An error that does not need to be corrected does not affect elements within the encoding space.

5 Stabilizer codes

5.1 The Generalized Pauli Group

The generalized Pauli group $\mathcal{G}_n$ is the set of operators

$$\mathcal{G}_n = \{g = \mu \mathbf{A}_1 \otimes \mathbf{A}_2 \otimes \mathbf{A}_3 \otimes ...\mathbf{A}_n\} \tag{18}$$

where $\mu \in \{1, -1, i, -i\}$ and each operator $\mathbf{A}_i \in \{\mathbf{I}_i, \mathbf{X}_i, \mathbf{Y}_i, \mathbf{Z}_i\}$ operates on a single qubit that is denoted with index i.

Properties of the Pauli group:

- Every element, when squared is either I or $-I$.
- Every pair of elements in the Pauli group either commutes or anti-commutes.

5.2 Weights and Quantum code distance

The weight t of a Pauli error is the number of nonidentity terms in its tensor product expression. For example an error $X \otimes I \otimes X \otimes I \otimes I$ has a weight of 2. The **weight** of an operator in the Pauli group is the number of non-identity elements in the tensor product.

As is the case for classical codes, the **distance** d of a quantum code is defined as the minimum size error that cannot be detected. For the 3 bit bit-flip error correcting code, a phase error in one of the qubits (e.g., Z_1) is not detected so we expect that $d = 1$.

For Shor's 9-bit error correcting code, all single qubit errors are detected and corrected so the distance d is at least 2. I think the Shor 9-bit code can also detect all errors that are distance two also so actually $d = 3$ for this code. This code will not detect all products of 3 single qubit errors. The bracket notation can be extended to include d with $[[n, k, d]]$. Shor's 9-bit error correcting code is $[[9, 1, 3]]$.

More formally, the **distance** of a quantum error correcting code is the minimum weight ($d > 0$), of an operator E in the the Pauli group such that the quantum error correcting code condition fails. In other words there an operator E (in the Pauli group), with weight d, for which

$$\langle v_a | E | v_v \rangle \neq c\delta_{ij}$$

(following equation 15) for two basis elements $|v_a\rangle, |v_b\rangle$ in the encoding (stabilized) space and some constant c.

5.3 Stabilizer Groups

We work in a 2^n dimensional Hilbert space for n qubits.

Definition: A **stabilizer group** S is a subgroup of the Pauli group G_n. The **vector space** V_S **stabilized** by **stabilizer** S is the set of vectors that are fixed by every element of the group S. The vector space V_S contains vectors $|v\rangle$ such that $g|v\rangle = |v\rangle$ for every element $g \in S$.

For example consider the group

$$S = \{I, Z_1Z_2, Z_2Z_3, Z_1Z_3\}$$

for a 3 qubit system. The operation $Z_1Z_2|000\rangle = |000\rangle$ so the state $|000\rangle$ is stabilized by Z_1Z_2. The subspace that is stabilized by Z_1Z_2 is spanned by the vectors $|000\rangle, |001\rangle, |110\rangle, |111\rangle$.

$$Z_1Z_2 \qquad \text{stabilizes} \qquad |000\rangle, |001\rangle, |110\rangle, |111\rangle.$$

The vector $|100\rangle$ is not stabilized by Z_1Z_2 as $Z_1Z_2|100\rangle = -|100\rangle$.

The subspace that is stabilized by Z_2Z_3 is spanned by $|000\rangle, |100\rangle, |011\rangle, |111\rangle$.

$$Z_2Z_3 \qquad \text{stabilizes} \qquad |000\rangle, |100\rangle, |011\rangle, |111\rangle.$$

The intersection of the two sets contains $|000\rangle, |111\rangle$. The vector space V_S stabilized by S is spanned by $|000\rangle, |111\rangle$.

$$\{I, Z_1Z_2, Z_2Z_3, Z_1Z_3\} \qquad \text{stabilizes} \qquad |000\rangle, |111\rangle. \tag{19}$$

Two conditions seem to be necessary for a nontrivial stabilizer in G_n, the Pauli subgroup for n qubits: The group cannot contain $-I$ and the elements of S must **commute**. If $-I$ were in the group $-I|v\rangle = -|v\rangle = |v\rangle$ giving $|v\rangle = 0$ which is a null space.

What are the possible non-trivial stabilizer groups for the 3 bit bit-flip error correction code encoded subspace? The encoded space C is spanned by $|000\rangle, |111\rangle$. The stabilizer is the same as discussed in the the previous example (equation 19).

Since a stabilizer group can contain many elements, we can more succinctly describe a stabilizer group with a set of elements that **generates** the group. We use the notation used by Chuang and Nielson. The group S generated by a set A denoted $\langle g_1 g_2 ... \rangle$ contains all elements g that can be written as $g = g_i g_j .....$ for $g_i, g_j .... \in A$. Here $g_1, g_2 ...$ are known as the **generators** of S.

Example: Find a stabilizer group that has stabilized subspace that is equal to the encoded subspace C, spanned by $|+++\rangle, |---\rangle$, for for the 3 bit phase-flip error correction code. The set $\langle X_1X_2, X_2X_3 \rangle$ is a stabilizer group that stabilizes subspace C. The generators generate two additional elements I, X_1X_3 so the entire group consists of $I, X_1X_2, X_2X_3, X_1X_3$. The operator Z_1Z_2 cannot be in the stabilizer group as $Z_1Z_2 |+++\rangle = |--+\rangle$.

$$\langle X_1X_2, X_2X_3 \rangle \qquad \text{stabilizes} \qquad |+++\rangle, |---\rangle . \tag{20}$$

We summarize the stabilizer groups, stabilized subspaces, which are the same thing as the encoding subspaces, for the 3-bit bit flip and 3-bit phase flip error correction codes.

Code	Stabilizer group generators		Basis for stabilized vector space
3-bit bit-flip	$\langle Z_1Z_2, Z_2Z_3 \rangle$	$Z\ Z\ I$ $I\ Z\ Z$	$\|000\rangle, \|111\rangle$
3-bit phase-flip	$\langle X_1X_2, X_2X_3 \rangle$	$X\ X\ I$ $I\ X\ X$	$\|+++\rangle, \|---\rangle$

In the above tables we are using a set of **independent** generators to describe the stabilizer space. By independent we mean that we choose a set that is large enough to describe the entire group but cannot be reduced in size. This means that each element in the generating set g_i cannot be written as a product of other elements in the generating set.

For the 3-bit phase flip code we have written X_1X_2 as XXI which is equivalent to $X \otimes X \otimes I$. This notation makes it clearer that the elements of the generating set are independent and commute.

What is a stabilizer group S for Shor's 9 bit encoded space? We need to find stabilizers of a space spanned by $\frac{1}{\sqrt{8}}(|000\rangle + |111\rangle)^{\otimes 3}$ and $\frac{1}{\sqrt{8}}(|000\rangle - |111\rangle)^{\otimes 3}$. $I, Z_1Z_2, Z_2Z_3, Z_1Z_3$ are in S as these flip 2 signs within the first cluster of 3 bits. Similarly Z_4Z_5, Z_5Z_6, Z_4Z_6 are in S these flip 2 signs within the second cluster of 3 bits. Similarly Z_7Z_8, Z_8Z_9, Z_7Z_9 are in S. The elements $X_1X_2X_3X_4X_5X_6$ and $X_4X_5X_6X_7X_8X_9$ are in S and so is their product $X_1X_2X_3X_7X_8X_9$. These operate on pairs of clusters of three bits. The stabilizer group for the 9-bit encoding space for Shor's 9 bit code is generated by

$$\langle Z_1Z_2, Z_2Z_3, Z_4Z_5, Z_5Z_6, Z_7Z_8, Z_8Z_9, X_1X_2X_3X_4X_5X_6, X_4X_5X_6X_7X_8X_9 \rangle$$

We can write the generating set for the Shor 9 bit code in the form of a table.

Stabilizer generators for the Shor 9 bit code	Basis for Stabilized space
$Z\ Z\ I\ I\ I\ I\ I\ I\ I$	
$I\ Z\ Z\ I\ I\ I\ I\ I\ I$	
$I\ I\ I\ Z\ Z\ I\ I\ I\ I$	
$I\ I\ I\ I\ Z\ Z\ I\ I\ I$	$\frac{1}{\sqrt{8}}(\|000\rangle + \|111\rangle)^{\otimes 3}$
$I\ I\ I\ I\ I\ I\ Z\ Z\ I$	$\frac{1}{\sqrt{8}}(\|000\rangle - \|111\rangle)^{\otimes 3}$
$I\ I\ I\ I\ I\ I\ I\ Z\ Z$	
$X\ X\ X\ X\ X\ X\ I\ I\ I$	
$I\ I\ I\ X\ X\ X\ X\ X\ X$	

Question: While a stabilizer group gives a stabilized subspace, does a subspace necessarily give a unique non-trivial stabilizer group (in the Pauli group)? A subspace of dimension m would be stabilized by operators in $U(m) \otimes U(n-m)$. Which are generated by a specific set of Pauli operators. I think the answer might depend on the dimension of the subspace. ....

5.4 Creating new stabilizers and stabilized subspaces via group conjugation

Consider a stabilizer group S and V_S the vector space stabilized by S. By definition of a stabilizer, for $|\psi\rangle \in V_S$, for any $g \in S$,

$$g\,|\psi\rangle = |\psi\rangle\,.$$

We apply a unitary operator U to $|\psi\rangle$

$$U\,|\psi\rangle = Ug\,|\psi\rangle = UgU^\dagger U\,|\psi\rangle\,.$$

This means that the state $U\,|\psi\rangle$ is stabilized by $UgU^\dagger$ for any $g \in S$ and any unitary operator U. The vector space UV_S is stabilized by the group

$$USU^\dagger = \{UgU^\dagger | g \in S\}.$$

This is relevant for the idea that non-encoding perpendicular subspaces are also stabilized by the stabilizer group.

5.5 Stabilizer codes

We take n to be the total number of qubits for the system in which encoding takes place and k is the number of qubits of information that we want to encode and project.

An $[[n,k]]$ **stabilizer code** $C(S)$ is

- A vector space V_S that is stabilized by a group S, known as the stabilizer group, that is a subgroup of the Generalized Pauli group $\mathcal{G}_n$.
- The stabilizer group S has $n-k$ independent and commuting generators;

$$S = \langle g_1, g_2, g_3 .... g_{n-k} \rangle.$$

- The stabilizer group S does not contain $-I$ (minus the identity operator).
- The stabilized vector space V_S is the same thing as the encoding space and is a subspace of the 2^n dimensional Hilbert space. The stabilized vector space has dimension k.

The three bit codes we discussed in section 2.1 and 2.2 and the 9-bit Shor code are examples of stabilizer codes.

Because all generators in the stabilizer commute, the stabilizer group is an 'abelian' group. All generators are their own inverse. A element $g \in S$ in the stabilizer group can be written in the following way

$$g = g_1^{i_1} g_2^{i_2} g_2^{i_2} .... g_{n-}^{i_{n-k}}$$

with indexes $i_1, i_2, ..., i_{n_k}$ that are either 1 or 0. Consequently the number of elements in the stabilizer group is

$$|S| = 2^{n-k}.$$

Question: Given k (number of qubits of information that we want to protect) and n (large space in which the encoding space lives) why must the number of independent generators for the stabilizer group be $n-k$?

Suppose we call n_g the number of generators. Because the stabilizer generators commute with each other, they can be simultaneously diagonalized. Because the generators are order 2, their eigenvalues must be ± 1. With n_g generators, there are 2^{n_g} possible measurements (each generator gives 2 possibilities). Because the generators are independent, all possible measurement values are possible. That means it must be possible to divide the full space into 2^{n_g} perpendicular subspaces. It is possible to show that each of these subspaces is stabilized by the stabilizer group and must have the same dimension as the stabilized or encoding subspace V_S. That means that 2^n should be equal to 2^{n_g+k} which implies that we desire the number of stabilizers $n_g = n-k$. If this is not true, then the stabilized subspace V_S will not have the desired dimension of the information we want to encode.

How are errors corrected with a stabilizer code? All members of the generalized Pauli group that are order 2 are Hermitian, so they can be used to make measurements! An error is corrected by performing measurements with the generators of the stabilizer group. Each measurement gives a single syndrome bit of information. You only get a

single bit of information from each measurement as the Pauli matrices, and the members of the generalized Pauli group have possible eigenvalues in the set $\{1, -1\}$.

Measurement of all generators of a stabilizer group S $g_1, g_2 ... g_{n-k}$ gives syndrome $\beta_1, \beta_2 ... \beta_{n-k}$. If no error occurred then $\beta_i = 1$ for all generators (for all indices i). This follows as $g_i |\psi\rangle = |\psi\rangle$, for state $|\psi\rangle \in V_S$ the vector space that is stabilized by stabilizer group S. Recall that V_S contains the encoded states!

If one or more of the syndrome bits gives a measurement of -1 instead of 1, then we know an error has occurred. The goal is to identify the error E from the syndrome measurements and then correct it by applying $E^\dagger$ to the state.

A particular error correcting code may only correct certain sets of possible errors.

Consider an element E, which we refer to as an error, which is an element of the generalized Pauli group $\mathcal{G}_n$. If $E \in S$ then it does not corrupt the code associated with stabilizer group S. There are two cases for $E \notin S$.

1) E commutes with all the elements of stabilizer S.

2) E anticommutes with one or more elements in S.

Suppose error E commutes with all elements in the stabilizer group S. For $|v\rangle \in V_S$ and with $Eg = gE$, we find that

$$\begin{aligned} Eg |v\rangle &= E |v\rangle \\ &= gE |v\rangle . \end{aligned}$$

This implies that $gE |v\rangle = E |v\rangle$ and this implies that $E |v\rangle$ would be in V_S. This implies that E operating on V_S (the stabilized vector space of S) is the same vector space, V_S. In other words for any $|v\rangle \in V_S$, the state $E |v\rangle$ is a vector in V_S. We say that E operating on V_S generates a vector space that is equal to V_S. However, E does not stabilize V_S. That means $E |v\rangle \in V_S$ for $|v\rangle \in V_S$ but it might be that $E |v\rangle \neq |v\rangle$. This type of error would be hard to identify! Data is corrupted by the error E.

However, if E anticommutes with an element g of the stabilizer group S, then $E |v\rangle$ is not in the vector space V_S for all $|v\rangle \in V_S$. This means that E applied to V_S generates a vector space that is perpendicular to V_S. If E anticommutes with g, $Eg = -gE$. This means that $-Eg |v\rangle = -E |v\rangle = gE |v\rangle$ for every $|v\rangle \in V_S$. Since g operating on $E |v\rangle$ gives $-E |v\rangle$, the vector $E |v\rangle \notin V_S$. This means that E operating on V_S generates a vector space that is perpendicular to V_S.

If E anticommutes with $g \in S$ then $gE |v\rangle = -Eg |v\rangle = -E |v\rangle$. This means that $E |v\rangle$ has eigenvalue of -1 for the operator g. If the state $E |v\rangle$ is measured with operator g, the measurement will give eigenvalue $\beta_g = -1$.

We take S a stabilizer group in $\mathcal{G}_n$ and $E \in \mathcal{G}_n$ that is not in S. A set of generators for S, is $\langle g_1, g_2, g_3 ... \rangle$. If operator E anticommutes with one of the elements $g \in S$, then E anticommutes with at least one of the generators of S. Equivalently if E commutes with all generators of S then it commutes with every element of S.[2]

[2]Proof: Suppose $g = g_i g_k ....$ anticommutes with E. This means that $E g_i g_k .. = -g_i g_k ... E$. Recall that

Suppose the encoded state is initially $|v\rangle$ and an error occurs giving a new state $E|v\rangle$. If E does not commute with an element in S then it does not commute with at least one of the generators of S. If we perform measurements with all the generators of S then one of them will reveal the error with a measurement value of -1.

In summary.

- The generators of a stabilizer group can be used for measurement. They have eigenvalues of ± 1 and so can be used to measure a syndrome!
- Measurements based on the generators of a stabilizer group will all give 1 if no error has occurred.
- At least one of the measurements of the generators of a stabilizer group will give -1 if a correctible error occurred. A correctible error is one that does not commute with at least one of the generators of the stabilizer group.

How is it possible to identify and correct more than one type of error? This is our next topic!

5.6 Normalizer and centralizers

The set of elements in a group G that commutes with the elements of a subgroup $H \subset G$ is known as the **centralizer** of the subgroup $Z(H)$ in G. In other words $g \in Z(H)$ if $gh = hg$ for al $h \in H$.

The **normalizer** $N(H)$ of a subgroup H is the set of all elements $g \in G$ such that $ghg^{-1} \in H$ for all $h \in H$.

It turns out that for the generalized Pauli group $\mathcal{G}_n$ any subgroup H that does not contain $-I$ has centralizer equal to normalizer; $N(H) = Z(H)$.[3]

every pair of elements in $\mathcal{G}_n$ either commutes or anticommutes. We can reorder every pair and slowly move E from the left to the right. To maintain the minus sign, we find that g must anticommute with an odd number of generators. We have shown the first half. The second half: We write an element in S in terms of the generators $g = g_i g_k .....$ We multiply by E giving $E g_i g_k ....$ If E commutes with all the generators then this must be equal to gE and so g must commute with all elements of S.

[3]Proof: Consider $c \in Z(H)$ the centralizer, then $ch = hc$ for all $h \in H$. We multiply by c^{-1} giving $h = c^{-1}hc$ which means that c is in the normalizer. All elements of the centralizer must be in the normalizer. Now consider $n \in N(H)$ the normalizer. By definition for all $h \in H$, the element $nhn^{-1} \in H$. Choose h. Multiply both sides by h'^{-1}, giving $nhn^{-1}h'^{-1} = I$. The elements h and n either commute or anti-commute as they are in the generalized Pauli group. Suppose that h, n anticommute, then we find that $hh'^{-1} = -I$. However this can't occur cause we insisted that $-I$ is not contained in H. Suppose that h, n commute. Then $h' = h$ and $nhn^{-1} = h$ which means that n is a member of the centralizer. We have shown that all elements of the normalizer are also elements of the centralizer. Hence the centralizer and normalizer must be the same group.

5.7 Examples of subspaces

For example, consider the stabilizer group S generated by $\langle X_1X_2, X_2X_3\rangle$ in a 3 qubit system. The operator iX_1 is in $\mathcal{G}_3$, and iX_1 commutes with all elements of S, so iX_1 is in the centralizer $Z(S)$ of stabilizer group S. However iX_1 is not in S. The operator iX_1 cannot be in S because its square is $-I$. The operator $X_1X_2X_3$, also in $\mathcal{G}_3$, is not in S but $X_1X_2X_3$ commutes with all elements in S and so it too is in $Z(S)$. The vector space V_S stabilized by S is spanned by $|+++\rangle, |---\rangle$. The operator $X_1X_2X_3$ does not stabilize $|+++\rangle$ as $X_1X_2X_3|+++\rangle = |---\rangle$. However $X_1X_2X_3$ which commutes with all elements of S, is not in the stabilizer even though $X_1X_2X_3$ operating on V_S gives back V_S.

How does a phase flip error Z_1 affect elements of stabilizer group S? The operator Z_1 anti-commutes with X_1X_2, so it sends V_S to a perpendicular subspace. $Z_1|+++\rangle \rightarrow |-++\rangle$ and $Z_1|---\rangle \rightarrow |+--\rangle$. The space spanned by $|-++\rangle, |+--\rangle$ is perpendicular to V_S which is spanned by $|+++\rangle, |---\rangle$.

Examples of operators that are not in a stabilizer group			
Stabilizer group $S = \langle X_1X_2, X_2X_3\rangle$	Basis for Stabilized space V_S $\|+++\rangle, \|---\rangle$		
Operator $E \notin S$	Commutes with S	Basis for EV_S	Correctible?
iX_1	yes	$\|+++\rangle, \|---\rangle$	No
$X_1X_2X_3$	yes	$\|+++\rangle, \|---\rangle$	No
Z_1	no	$\|-++\rangle, \|+--\rangle$	Yes
Z_1Z_3	no	$\|-+-\rangle, \|+-+\rangle$	Yes

5.8 Syndromes using stabilizer generators for 3-bit codes and Shor's 9-bit code

An operator, E, which is an element in the generalized Pauli group $\mathcal{G}_n$ we refer to as an error. An error that commutes with all generators of a stabilizer group S but is not in the stabilizer group is **not correctible** by measurements of these generators. An error that anti-commutes with one generator is correctible.

Suppose we have a state $|v\rangle \in V_S$, the space stabilized by S. Because $|v\rangle \in V_S$ it is a possible error free encoded state. We consider measurements by the generators of S if a specific error occurs. In other words, we list the eigenvalues of stabilizer elements for a state $E|v\rangle$. The eigenvalue is either 1 or -1. It is 1 if E commutes with the stabilizer generator, and it is -1 if it anti-commutes with the stabilizer generator.

Each possible error generated subspace (the subspace that is E operating on V_S) should have a unique signature that is determined through measurement of the stabilizer generators. This allows an operator to be applied that corrects the error.

Syndrome for the 3-bit bit flip error code		
	Stabilizer generators	
Error	Z_1Z_2	Z_2Z_3
X_1	-1	1
X_2	-1	-1
X_3	1	1

The above table lists what would be measured from the stabilizer generators for the 3-bit bit flip error code with stabilizer $\langle Z_1Z_2, Z_2Z_3\rangle$. How did we fill this table? We determined whether the error commutes or anticommutes with the stabilizer generator. If it commutes, the relevant eigenvalue would be 1 otherwise it would be -1.

The above table lists all possible single bit bit-flip errors. Each one has a different set of syndrome measurements. This means that each of the single bit flip errors X_1, X_2, X_3 is correctible. For example if $-1, -1$ is measured then we could correct the error by applying X_2.

The set of errors $\{X_1, X_2, X_3\}$ is **correctible** by the 3 bit bit flip error code with stabilizer $\langle Z_1Z_2, Z_2Z_3\rangle$.

Let us consider the set of 2 bit flip errors.

Syndrome for the 3-bit bit flip error code		
	Stabilizer generators	
Error	Z_1Z_2	Z_2Z_3
X_1X_2	1	-1
X_2X_3	-1	1
X_1X_3	-1	-1

The set of errors $\{X_1X_2, X_2X_3, X_1X_3\}$ is **correctible** by the 3 bit bit flip error code with stabilizer $\langle Z_1Z_2, Z_2Z_3\rangle$.

However if we consider the set $\{X_1, X_2, X_3, X_1X_2, X_2X_3, X_1X_3\}$ which contains both single and double bit-flip errors, the set of errors is **not** correctible. This is because there are only 4 possible syndrome measurements for a stabilizer group with 2 generators and the set contains 6 possible errors. Because single bit phase flip errors Z_1, Z_2, Z_3 commute with the stabilizers, you cannot detect phase-flip errors with the code generated by $\langle Z_1Z_2, Z_2Z_3\rangle$.

For the 3-bit phase flip error code has stabilizer generators $\langle X_1X_2, X_2X_3\rangle$, we we can similarly determine which single bit operators commute with each generator to construct a syndrome for single bit phase flip error correction.

Syndrome for the 3-bit phase flip error code		
	Stabilizer generators	
Error	X_1X_2	X_2X_3
Z_1	-1	1
Z_2	-1	-1
Z_3	1	1

The set of errors $\{Z_1, Z_2, Z_3\}$ is correctable for the 3-bit code with stabilizer generators $\langle X_1X_2, X_2X_3\rangle$.

Let's make a similar table for the Shor 9-bit code

Error	Generator measurement for 9-bit code							
	Z_1Z_2	Z_2Z_3	Z_4Z_5	Z_5Z_6	Z_7Z_8	Z_8Z_9	$X_1X_2X_3$ $X_4X_5X_6$	$X_4X_5X_6$ $X_7X_8X_9$
X_1	-1	1	1	1	1	1	1	1
X_2	-1	-1	1	1	1	1	1	1
X_3	1	-1	1	1	1	1	1	1
X_4	1	1	-1	1	1	1	1	1
X_5	1	1	-1	-1	1	1	1	1
X_6	1	1	1	-1	1	1	1	1
X_7	1	1	1	1	-1	1	1	1
X_8	1	1	1	1	-1	-1	1	1
X_9	1	1	1	1	1	-1	1	1
$Z_1Z_2Z_3$	1	1	1	1	1	1	-1	1
$Z_4Z_5Z_6$	1	1	1	1	1	1	-1	-1
$Z_7Z_8Z_9$	1	1	1	1	1	1	1	-1

The Shor 9-bit code has stabilizer with 8 generators. We could conceivably have 2^8 possible measurement values. The Shor 9-bit code is often called degenerate. It is possible to devise a more efficient stabilizer code that uses fewer than 9 bits and yet can still correct for all single qubit errors.

5.9 Conditions for error correcting in Stablizer codes

Let S be a stabilizer group for stabilizer code $C(S)$. Let the set $S_E = \{E_i\}$ be a set of operators in the Pauli group $\mathcal{G}_n$.

$$\text{If} \quad E_j^\dagger E_k \notin N(S) - S \qquad \text{for all } j,k \tag{21}$$

then $\{E_i\}$ is a **correctable set of errors** for $C(S)$.

What is meant by $N(S) - S$? We mean the set of elements in $\mathcal{G}_n$ that are in the normalizer of S but not in S itself. That means $E_j^\dagger E_k$ could be in the stabilizer S. This condition (in equation 21) is similar to that we discussed previously in equation 15.

Equation 21 implies that either all pairs $E_i^\dagger E_j$

- are in the stabilizer
- or they anticommute with an element in the stabilizer.

To show that a set of errors is correctible, you need to show that these conditions are satisfied.

If $E_i^\dagger E_j \in N(S)$ then $E_i^\dagger E_j$ commutes with all elements in S. (This is because the normalizer of any stabilizer group is equivalent to its centralizer, and the centralizer is the set of elements that commute with the subgroup). We consider two cases:

1) If $E_i^\dagger E_j \notin S$ This means that $E_i |v_a\rangle$ is not necessarily the same as $E_j |v_a\rangle$. That means if we use E_i to correct an error caused by E_j we could corrupt the encoded state-vector. A correctible set of errors should not contain any pairs like this!

2) If $E_i^\dagger E_j \in S$ then an error by E_j is corrected by E_i. An encoded state-vector would be returned to its original state because S is a stabilizer. A correctible set of errors could contain pairs like this!

If $E_i^\dagger E_j \notin N(S)$ then $E_i^\dagger E_j$ anti-commutes with at least one generator of the stabilizer S. E_i and E_j generate orthogonal subspaces when operating on V_S and if measurements are made with the generators of the stabilizer then it should be possible to differentiate between the two errors.

Hence the condition for a set of errors to be correctible in equation 21 is consistent and equivalent to that we wrote previously in equation 15.

As we established in section 3.2 the Shor 9-bit code corrects all single qubit errors. Let's check that the set of all single qubit errors obey equation 15 for the Shor 9-bit code.

The set of single qubit errors is

$$S_E = \{X_1, Y_1, Z_1, X_2, Y_2, Z_2, ...X_9, Y_9, Z_9\}. \tag{22}$$

We check that this set of errors satisfies equation 15 or 21.

Consider products of operators operating on a single qubit. As we can form Y via iXZ, a product of two elements of the error list S_E, with both operating on a single qubit, gives an element proportional to an error operator on a single qubit. We check that none of the single qubit operators commute with every generator of the stabilizer group. For example: X_1, Y_1, X_2 and Y_2 do not commute with Z_1Z_2. X_3, Y_3 do not commute with Z_2Z_3. Z_1, Z_2 and Z_3 do not commute with $X_1X_2X_3X_4X_5X_6$.

Now we consider products of two operators in the set S_E (equation 22) where each operator operates on a different qubit. Products like Z_1Z_3 are in the stabilizer group. The operators Z_1, Z_4 give perpendicular subspaces when operating on the encoding basis, $|\tilde{0}\rangle, |\tilde{1}\rangle$,

so will satisfy equation 15. Equivalently Z_1Z_4 anticommutes with $X_1X_2X_3X_4X_5X_6$ so is not in the normalizer of the stabilizer group.

The product X_1X_4 does not commute with Z_1Z_2 so is not in the normalizer. The product of any two X operators operating on two different qubits will not be in the normalizer.

Lastly any X and any Z but operating on two different qubits will give perpendicular subspaces. A products: X_iZ_j with $i \neq j$ does not commute with Z_iZ_k (for some k).

Using equation 21 we have roughly shown that the set of all single qubit operators in the generalized Pauli group is correctable with the 9-bit Shor stabilizer code.

Interestingly I think we can expand the list of errors to also include all two-qubit operators. Apparently the 9-bit Shor stabilizer code can also correct all two qubit errors.

5.10 On degeneracy

Consider equation 15 which determines if a set of errors is correctable and which we repeat here! For all errors E_i, E_j in a set of correctable errors S_E

$$\langle v_a| E_i^\dagger E_j |v_b\rangle = m_{ij}\delta_{ab}$$

for any basis vectors $|v_a\rangle, |v_b\rangle \in C$.

If

$$\langle v_a| E_i^\dagger E_j |v_b\rangle = 0 \quad \text{for all} \quad |v_a\rangle, |v_b\rangle \in C \tag{23}$$

for all $E_i \neq E_j$, with $E_i, E_j \in S_E$, a set of correctable errors, then the code $C(S)$ is **not degenerate**.

If a quantum code encodes k qubits in an n qubit space, the encoded information has dimension 2^k. That means that the stabilized subspace C has dimension 2^k. It lives in a Hilbert space that is dimension 2^n. Each error E_i when operating on C gives a subspace perpendicular to C, the stabilized space, that has the same dimension as C. If each error gives an independent orthogonal subspace (via E_iC) (independent of the other errors) then there can be at most 2^{n-k} correctable errors. If equation 23 is satisfied, and the code is not degenerate, then there are at most most 2^{n-k} correctable errors.

The Shor 9-bit code is degenerate because Z_1, Z_2, Z_3 all generate the same subspace when operating on C. For example

$$\frac{1}{\sqrt{8}}(\langle 000| + \langle 111|)^{\otimes 3} Z_1Z_2 \frac{1}{\sqrt{8}}(|000\rangle + |111\rangle)^{\otimes 3} = 1.$$

There are pairs of correctable errors that do not satisfy equation 23.

5.11 A non-generate 5 bit code

It is possible to make a code that is more efficient than the Shor 9-bit code that still corrects all single qubit errors. Such a code is generated by the following generators:

	Generators for a 5-bit code
g_1	$X\ Z\ Z\ X\ I$
g_2	$I\ X\ Z\ Z\ X$
g_3	$X\ I\ X\ Z\ Z$
g_4	$Z\ X\ I\ X\ Z$

The stabilizer group S generated by these four generators has 16 elements. There are 4 generators. Each element is its own inverse. The group contains the identity. Each pair of generators gives another element in the stabilizer group ($(4 \times 3)/2 = 6$ more elements). Each triplet of generators gives another element (4 more elements in the stabilizer group). There is a unique element that is a product of all four generators. The total number of elements in the stabilizer group is $1 + 4 + 6 + 4 + 1 = 16$. Equivalently there are $2^4 = 16$ ways to choose products of the generators.

5.12 Finding the encoding subspace from the stabilizer group

Starting with a stabilizer group, S, can we find the encoding subspace? Equivalently, how can we find a basis for the encoding subspace C? For example, what does the encoding subspace look like for the 5-bit code?

Starting with $|\psi_0\rangle$ we can construct the state

$$|\Psi_0\rangle = \frac{1}{\sqrt{|S|}} \sum_{g \in S} g\,|\psi_0\rangle \tag{24}$$

where $|S|$ is the number of elements in the stabilizer group. Note this is a sum over all the elements, not just a sum over the generators of the stabilizer group. This will give us a single basis element for C. The state $|\Psi_0\rangle$ is in the stabilized vector space (or C) because if you operate on it by any element in the stabilizer, the elements in the sum in equation 24 will just simply be rearranged or permuted. (This follows as gh_1 cannot be equal to gh_2 if $h_1 \neq h_2$ for g, h_1, h_2 elements of a group).

For the 5-bit stabilizer code, the encoding space can be constructed with

$$\begin{aligned} |0\rangle \rightarrow \left|\tilde{0}\right\rangle &= \frac{1}{4} \sum_{g_i \in S} g_i\,|00000\rangle \\ |1\rangle \rightarrow \left|\tilde{1}\right\rangle &= \frac{1}{4} \sum_{g_i \in S} g_i\,|11111\rangle\,, \end{aligned}$$

where $1/4$ follows from the 16 members of the group.

How can we be sure that we have two perpendicular states? We notice that $\bar{Z} = ZZZZZ$ is not in the stabilizer group but does commute with the stabilizer group. The state $\left|\tilde{0}\right\rangle$ is an eigenstate of $\bar{Z}$ with eigenvalue 1. We operate on $\left|\tilde{1}\right\rangle$ with $\bar{Z}$

$$\begin{aligned}
\bar{Z}\left|\tilde{1}\right\rangle &= \frac{1}{4}\sum_{g_i \in S} \bar{Z} g_i \left|11111\right\rangle \\
&= \frac{1}{4}\sum_{g_i \in S} g_i \bar{Z} \left|11111\right\rangle \quad \text{because } \bar{Z} \text{ commutes with } S \\
&= \frac{1}{4}\sum_{g_i \in S} g_i \left|11111\right\rangle \times -1 \\
&= -\left|\tilde{1}\right\rangle.
\end{aligned}$$

We find that $\left|\tilde{1}\right\rangle$ is also an eigenstate of $\bar{Z}$ but with a different eigenvalue, -1, so it must be perpendicular to $\left|\tilde{0}\right\rangle$.

Note $\bar{X} = XXXXX$ also commutes with the stabilizer. Denote U as the encoding operation.

$$\begin{aligned}
\bar{X}\left|\tilde{0}\right\rangle &= \bar{X}U\left|00000\right\rangle \\
&= U\bar{X}\left|00000\right\rangle \\
&= U\left|11111\right\rangle \\
&= \left|\tilde{1}\right\rangle
\end{aligned}$$

This means that $\bar{X}$ on the encoding space is equivalent to a logical NOT operation. This will become relevant when we discuss fault tolerant logical operations.

Why do discuss the encoding space in terms of operators that commutes with (but are not in) the stabilizer group? An operator h that commutes with all stabilizer generators, but is not in the stabilizer would cause an undetectable error if $h\left|\tilde{0}\right\rangle \notin V_S$ where V_S is the stabilized subspace. So we require that $h\left|\tilde{0}\right\rangle \in V_S$. The operator h can be used to create a state perpendicular to $\left|\tilde{0}\right\rangle$ (the $\bar{X}$ operator) or we could use it to generate a basis for the encoding subspace (like the $\bar{Z}$ operator).

5.13 The Quantum Hamming Bound

Let t be the maximum weight for which the set of Pauli group elements of weight t or less is correctable by a non-degenerate $[[n, k]]$-quantum code. In other words, the code cannot correct errors with weight greater than t. In section 5.10 we argued that a non-degenerate code could correct at most 2^{n-k} errors. The number of errors of weight 1 is $3n$ as we can choose any of the 3 Pauli matrices for each qubit. The number of errors of weight 2 is $9n(n-1)/2$. This corresponds to n ways to chose which qubit gets one Pauli matrix followed by $n-1$ ways to choose which qubit gets the second Pauli matrix. Once the two

qubits are chosen, there are 3 possible Pauli matrices to chose from for each qubit. Why do we divide by 2? I think this is because Z_1X_2 can be converted to X_1Z_2 using single qubit operations. Using binomial coefficients,[4] the number of possible errors of weight t is $3^t \binom{n}{t}$.

Any non-degenerate $[[n, k]]$-quantum code that corrects all errors with weight t or less must satisfy the **quantum Hamming bound**

$$\sum_{i=0}^{t} 3^i \binom{n}{t} \leq 2^{n-k}. \tag{25}$$

Why does the sum start with 0? I think this is because we need to remember to count the space C itself.

Non-degenerate codes are less efficient than degenerate codes (which can correct more errors because sometimes one operation can correct more than one type of error) which is why the bound is given for non-degenerate codes.

Let's look at the 5-bit code. $n = 5, k = 1$ and $2^{n-k} = 2^4 = 16$. In principle we could correct 16 errors. With 4 generators we have $2^4 = 16$ syndrome possible measurements, and that's consistent with 15 correctable errors (included an identify for no error correction needed). With 5 bits, there are 15 possible single qubit errors in the generalized Pauli group (this is 3×5). If we include the identity then 16 equals the quantum Hamming bound.

6 CSS Codes

CSS stands for Calderbank, Shor and Steane. CSS codes are a subclass of stabilizer quantum error correcting codes. Two classical error correcting codes C_1, C_2, that are $[n, k_1], [n, k_2]$ are used to construct a quantum error correcting code $[[n, k_1 - k_2]]$. Apparently the parity check matrix of the first classical code is used as a generator matrix for the second, so the two classical codes are dual or perpendicular to each other. The duality condition can be related to the fact that phase flip errors can be related to bit flip relations via the duality condition $Z_i = HX_iH$, where i is for each qubit. The most famous of the CSS quantum codes is Steane's quantum [[7,1]] error correcting code.

6.1 Generator and parity matrices in classical codes

Because CSS codes are built from two dual classical codes, we introduce Hamming codes.

Classical linear codes are described in terms of generator and parity check matrices.

[4] The binomial coefficient $\binom{n}{t} = \frac{n!}{t!(n-t)!}$

If we take the initial state and write it as a vertical vector then a generator matrix G can be used to describe the encoding operation. Because it involves multiplying a vector by a matrix, the code can be described as linear.

The generator function for the classical three bit repetition error correcting code is

$$G_{3,repetition} = \begin{pmatrix} 1 \\ 1 \\ 1 \end{pmatrix} \tag{26}$$

and encoding gives

$$\begin{pmatrix} 1 \\ 1 \\ 1 \end{pmatrix} (0) = \begin{pmatrix} 0 \\ 0 \\ 0 \end{pmatrix} \qquad \begin{pmatrix} 1 \\ 1 \\ 1 \end{pmatrix} (1) = \begin{pmatrix} 1 \\ 1 \\ 1 \end{pmatrix}$$

A parity check matrix is used to detect errors in a classical error correcting code. An example of a parity check matrix for the 3-bit repetition code is

$$H_{3,repetition} = \begin{pmatrix} 1 & 1 & 0 \\ 0 & 1 & 1 \end{pmatrix}.$$

Each row contains two 1s. If you multiply the matrix by a three bit state, and sums are done modulo 2, then the matrix checks the parity of the first two and second two bits.

We do arithmetic mod 2 in matrix multiplication. The parity matrix times encoded states

$$H \cdot (000) = \begin{pmatrix} 1 & 1 & 0 \\ 0 & 1 & 1 \end{pmatrix} \begin{pmatrix} 0 \\ 0 \\ 0 \end{pmatrix} = \begin{pmatrix} 0 \\ 0 \end{pmatrix}$$

$$H \cdot (111) = \begin{pmatrix} 1 & 1 & 0 \\ 0 & 1 & 1 \end{pmatrix} \begin{pmatrix} 1 \\ 1 \\ 1 \end{pmatrix} = \begin{pmatrix} 0 \\ 0 \end{pmatrix}.$$

What if there is an error in the first bit?

$$H \cdot (100) = \begin{pmatrix} 1 & 1 & 0 \\ 0 & 1 & 1 \end{pmatrix} \begin{pmatrix} 1 \\ 0 \\ 0 \end{pmatrix} = \begin{pmatrix} 1 \\ 0 \end{pmatrix}$$

$$H \cdot (011) = \begin{pmatrix} 1 & 1 & 0 \\ 0 & 1 & 1 \end{pmatrix} \begin{pmatrix} 0 \\ 1 \\ 1 \end{pmatrix} = \begin{pmatrix} 1 \\ 0 \end{pmatrix}.$$

Similarly if there is an error in the second bit

$$H \cdot (010) = \begin{pmatrix} 1 & 1 & 0 \\ 0 & 1 & 1 \end{pmatrix} \begin{pmatrix} 0 \\ 1 \\ 0 \end{pmatrix} = \begin{pmatrix} 1 \\ 1 \end{pmatrix}$$

$$H \cdot (101) = \begin{pmatrix} 1 & 1 & 0 \\ 0 & 1 & 1 \end{pmatrix} \begin{pmatrix} 1 \\ 0 \\ 1 \end{pmatrix} = \begin{pmatrix} 1 \\ 1 \end{pmatrix}.$$

Similarly if there is an error in the third bit

$$H \cdot (001) = \begin{pmatrix} 1 & 1 & 0 \\ 0 & 1 & 1 \end{pmatrix} \begin{pmatrix} 0 \\ 0 \\ 1 \end{pmatrix} = \begin{pmatrix} 0 \\ 1 \end{pmatrix}$$

$$H \cdot (110) = \begin{pmatrix} 1 & 1 & 0 \\ 0 & 1 & 1 \end{pmatrix} \begin{pmatrix} 1 \\ 1 \\ 0 \end{pmatrix} = \begin{pmatrix} 0 \\ 1 \end{pmatrix}.$$

The error is uniquely identified.

The parity check matrix gives the syndrome measurement.

The product Hs (where s is the code word) gives a two dimensional vector and there are four possible such vectors (00, 01, 10, 11). This means the parity check matrix can detect three correctible errors, as 00 corresponds to no error. The parity matrix can detect the three possible single bit errors, but not errors in two bits simultaneously.

To ensure that the parity matrix gives a zero vector on an error free encoded state, the product of the parity check and generator matrix should give a zero matrix;

$$HG = 0.$$

6.2 The classical [7,4] Hamming code

The classical [7,4] Hamming code takes 4 bits and encodes them as 7 bits. The **generator matrix** is

$$G_{[7,4]} = \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \\ 1 & 1 & 1 & 0 \\ 0 & 1 & 1 & 1 \\ 1 & 0 & 1 & 1 \end{pmatrix}. \tag{27}$$

Note that the first 4 rows are the identity matrix and the last three rows look like parity checks for triplets of bits. The encoded words have the property that any pair differs in at least 3 bits. This is because each column of the generator matrix has 3 ones in it.

The **parity matrix** matching the generator matrix of equation 27 is

$$H_{[7,4]} = \begin{pmatrix} 1 & 1 & 1 & 0 & 1 & 0 & 0 \\ 0 & 1 & 1 & 1 & 0 & 1 & 0 \\ 1 & 0 & 1 & 1 & 0 & 0 & 1 \end{pmatrix}. \tag{28}$$

The columns are the 7 non-zero three bit strings. Each row contains an even number of 1s and performs a parity check from 4 bits. Notice that the right three rows gives a 3×3 dimensional identity matrix. The remaining part of the matrix is the same as the bottom four rows of the generator matrix. Using a 3×4 matrix A, we can write

$$\begin{aligned} G_{[7,4]} &= \begin{bmatrix} I_4 \\ A \end{bmatrix} \\ H_{[7,4]} &= \begin{bmatrix} A & I_3 \end{bmatrix}. \end{aligned} \tag{29}$$

Here I_4 is the 4×4 identity and I_3 is a 3×3 identity matrix. If you multiply $H_{[7,4]}G_{[7,4]}$ you can see that you wind up adding A to itself and this has to give zero, no matter what A is.

$$H_{[7,4]}G_{[7,4]} = \begin{pmatrix} 1 & 1 & 1 & 0 & 1 & 0 & 0 \\ 0 & 1 & 1 & 1 & 0 & 1 & 0 \\ 1 & 0 & 1 & 1 & 0 & 0 & 1 \end{pmatrix} \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \\ 1 & 1 & 1 & 0 \\ 0 & 1 & 1 & 1 \\ 1 & 0 & 1 & 1 \end{pmatrix} = \begin{pmatrix} 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \end{pmatrix} \tag{30}$$

6.3 A perpendicular [7,3] Hamming code

From the [7,4] Hamming code, we construct what is known as a perpendicular or dual code.

For the [7,4] code the parity check matrix in equation 28 is 3×7 (rows × columns). We take the transpose of this matrix to construct a generator matrix that is 7×3 giving a [7,3] code, $G_{[7,3]} = H^T_{[7,4]}$.

For the [7,4] code the generator matrix in equation 27 is 7×4 (rows × columns). We take the transpose of this matrix and create a parity matrix from it that is 4×7; $H_{[7,3]} = G^T_{[7,4]}$.

Because the [7,4] generator and parity check matrices were in the form of equation 29 the [7,3] generator and parity matrices also satisfy $HG = 0$.

The two dual codes can be used to construct a quantum [[7,1]] quantum error correcting code, known as the Steane's code.

6.4 Constructing a CSS code

We start with two perpendicular codes C_1, C_2 with $C_2 \in C_1$. Let C_1 and C_2 refer to classical code $[n, k_1]$ and $[n, k_2]$, respectively. A code word $x \in C_1$ (that has n bits) can be

used to define a state

$$|x + C_2\rangle = \frac{1}{\sqrt{|C_2|}} \sum_{y \in C_2} |x + y\rangle \,. \tag{31}$$

This means we get a state for each code word in C_1 that depends on C_2. Here $|C_2|$ is the number of code words in C_2. The resulting quantum code is $[[n, k_1 - k_2]]$.

For C_1 the [7,4] Hamming code, encoded words, referred to as x, have 7 bits. For C_2 the [7,3] code perpendicular to it, code words $y \in C_2$ also have 7 bits. The expression 31 gives 7 bit code words and the associated quantum code has $n = 7$ bits. However, not all $x \in C_1$ give unique encoded quantum states, (referred to confusingly as $|x + C_2\rangle$). If $x' = x + z$ with $z \in C_2$ then equation 31 implies that $|x + C_2\rangle = |x' + C_2\rangle$. As C_1 encodes 4 bits and C_2 encodes 3 bits, C_1 has twice as many encoded words. The resulting quantum code is [[7,1]].

6.5 Stabilizer group for Steane's 7 bit code

The 7-bit code constructed via 2 perpendicular classical Hamming codes is also a stabilizer code. The stabilizer group for the CSS Steane code is:

	Generators for the 7-bit code
g_1	$I\ I\ I\ X\ X\ X\ X$
g_2	$I\ X\ X\ I\ I\ X\ X$
g_3	$X\ I\ X\ I\ X\ I\ X$
g_4	$I\ I\ I\ Z\ Z\ Z\ Z$
g_5	$I\ Z\ Z\ I\ I\ Z\ Z$
g_6	$Z\ I\ Z\ I\ Z\ I\ Z$

The dual structure of Steane's 7-bit code is related to the similarity of g_1, g_2, g_3 and g_4, g_5, g_6 generators. These sets of generators can be considered dual to one another via $X \to Z$. The structure of the parity check matrices is related to the structure of the generators. **How?** The order of the columns for $H_{[7,4]}$ (in equation 28) can be rearranged to give g_1, g_2, and g_3. Then g_4, g_5, g_6 are constructed from g_1, g_2, g_3 by converting X to Z.

The code words are generated from equation 31 or by considering the stabilizer group

based on the 6 generators.

$$\begin{aligned}\left|\tilde{0}\right\rangle =&\frac{1}{\sqrt{8}}(|0000000\rangle + |1110100\rangle + |1101010\rangle + |0011110\rangle \\ &+ |1011001\rangle + |0101101\rangle + |0110011\rangle + |1000111\rangle) \\ \left|\tilde{1}\right\rangle =&\frac{1}{\sqrt{8}}(|1111111\rangle + |0001011\rangle + |0010101\rangle + |1100001\rangle \\ &+ |0100110\rangle + |1010010\rangle + |1001100\rangle + |0111000\rangle). \end{aligned} \tag{32}$$

The two encoding states satisfy $\left|\tilde{0}\right\rangle = X_1X_2X_3X_4X_5X_6X_7\left|\tilde{1}\right\rangle$.

As there are 6 generators, the syndrome detects $2^6 = 64$ possible errors which includes the 14 single qubit bit flip and phase flip errors.

7 Fault Tolerant Computing

We have discussed encoding so that a quantum state is protected from error. We now discuss the protection of quantum information as it dynamically undergoes computation. The goal is to keep the probability of error per gate below a certain threshold. To do a computation on error protected data, we need to perform the computation directly on encoded states. We have previously discussed how to replace each qubit in the original circuit with an encoded block of qubits. We now discuss how to replace each **operation** or gate with a fault tolerant version. Essentially we need *encoded gates*. A badly encoded gate can cause an error to propagate.

An example of an error *propagating* is shown in Figure 9. In Figure 9 the top circuit shows the desired operation. If an error X_1 occurs before the CNOT then the operation is X_1 CNOT. However the effective action is CNOT X_1X_2, as shown in the bottom circuit. This implies that

$$\begin{aligned} X_1 \text{ CNOT} &= \text{CNOT } X_1X_2 \\ \text{CNOT } X_1 \text{ CNOT} &= X_1X_2 \end{aligned}$$

where I am using the fact that CNOT is its own inverse. If I define U as the unitary operation for CNOT then

$$U^{-1}X_1U = X_1X_2.$$

This type of conjugation gives us a way of discussing how errors *propagate* through a circuit.

7.1 Operations on encoded states

How do we perform common quantum gates on error encoded states?

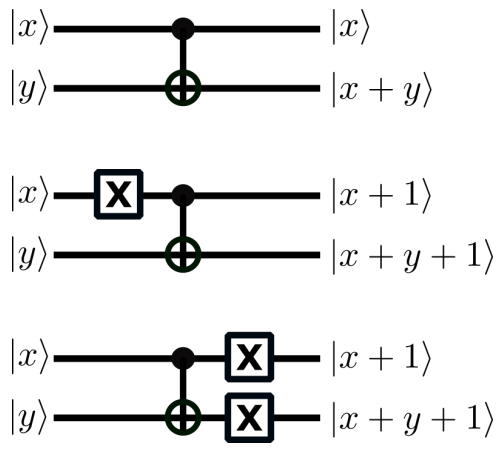

Figure 9: The intended gate is a CNOT as shown on in top circuit. If bit flip error occurs on the top bit, as shown with the Pauli X in the middle circuit, both the top and bottom bits suffer a bit flip error, as shown in the bottom circuit. We say that the error is *propagated* by the CNOT gate.

Consider Shor's 9-bit code,

$$\begin{aligned}\left|\tilde{0}\right\rangle &= \frac{1}{\sqrt{8}}(|000\rangle + |111\rangle)^{\otimes 3} \\ \left|\tilde{1}\right\rangle &= \frac{1}{\sqrt{8}}(|000\rangle - |111\rangle)^{\otimes 3}.\end{aligned}$$

How do we describe an X or Z operation for unencoded qubit but in the encoded space?

$$\begin{aligned}Z_1Z_4Z_7\left|\tilde{0}\right\rangle &= \frac{1}{\sqrt{8}}(|000\rangle - |111\rangle)^{\otimes 3} \\ &= \left|\tilde{1}\right\rangle \\ Z_1Z_4Z_7\left|\tilde{1}\right\rangle &= \frac{1}{\sqrt{8}}(|000\rangle + |111\rangle)^{\otimes 3} \\ &= \left|\tilde{0}\right\rangle.\end{aligned}$$

The operator $Z_1Z_4Z_7$ or equivalently the operator $Z_1Z_2Z_3Z_4Z_5Z_6Z_7Z_8Z_9$ acts in the 9-bit encoding space equivalently to X in the 1 bit original space.

The operator $\bar{Z} = Z^{\otimes 9}$ is the logical operator in the 9-bit encoding space that is equivalent to X in the 1 bit original space.

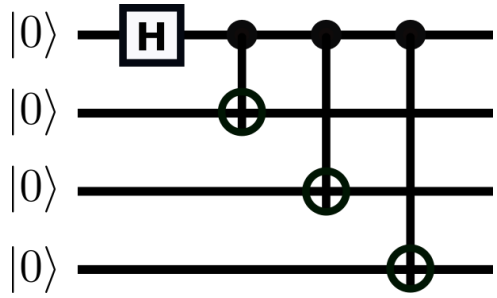

Figure 10: A way to construct a cat state that is not fault tolerant. The resulting cat state is $\frac{1}{\sqrt{2}}(|0000\rangle + |1111\rangle)$.

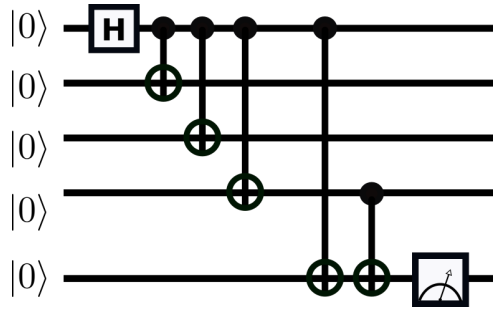

Figure 11: A way to find out if the contruction of the cat state has been compromised. If the top and bottom qubit are not the same then the measurement would be 1 instead of 0. In this case the cat state should be discarded and constructed again!

$$\begin{aligned} X_1X_2X_3X_4X_5X_6X_7X_8X_9\left|\tilde{0}\right\rangle &= \frac{1}{\sqrt{8}}(|000\rangle + |111\rangle)^{\otimes 3} \\ &= \left|\tilde{0}\right\rangle \\ X_1X_2X_3X_4X_5X_6X_7X_8X_9\left|\tilde{1}\right\rangle &= -\frac{1}{\sqrt{8}}(|000\rangle - |111\rangle)^{\otimes 3} \\ &= -\left|\tilde{1}\right\rangle. \end{aligned}$$

The operator $\bar{X} = X^{\otimes 9}$ the logical operator in the 9-bit encoding space that is equivalent to Z in the 1 bit original space.

The operators $\bar{Z}$ and $\bar{X}$ commute with the generators of the stabilizer group for the 9-bit code.

7.2 Fault tolerant NOT and Z gates for the 9 bit code

Fault tolerant NOT:

Let's use the 9-bit code again as an example. We showed in the previous section that applying a NOT (an X) gate to a single qubit can be performed in the encoding space by applying $\bar{Z}$ (Z-gates to all 9 bits) within the encoding space.

Suppose a single qubit error E_i occurs before the $\bar{Z}$ is performed. The encoded state $|\psi\rangle$ becomes $\bar{Z}E|\psi\rangle$. We then perform an error correction step. We check to see if the syndrome detection and error correction steps restore the state to the desired one.

We first consider a Z_i error in any qubit. The two operators, $\bar{Z}$ and Z_i commute, so it does not matter if the error occurs before or after the $\bar{Z}$ operation. The state $\bar{Z}|\psi\rangle$ lies in the encoded or stabilized subspace. But $\bar{Z}Z_i|\psi\rangle = Z_i\bar{Z}|\psi\rangle$ does not lie in the encoded subspace. That means the error will be detected by the error detection syndrome. Furthermore the error that is detected will be a single bit Z error that will appropriately corrected.

If the error is an X_i error in any qubit, the two operators $\bar{Z}$ and X_i anti-commute. The states $\bar{Z}X_i|\psi\rangle = -X_i\bar{Z}|\psi\rangle$ do not lie in the encoded subspace. The single qubit error will be detected in both cases and X_i applied to correct it. If the error occurs prior to the application of $\bar{Z}$, the corrected state would be $X_i\bar{Z}X_i|\psi\rangle = -\bar{Z}|\psi\rangle$. If the error occurs after application of $\bar{Z}$, the correct state would be $X_iX_i\bar{Z}|\psi\rangle = \bar{Z}|\psi\rangle$. Ignoring global phase, the error correction procedure would work.

Previously we showed that if you can correct for X and Z errors, then all single qubit errors would be corrected.

Fault tolerant Z:

What about the Z gate? Previously we showed that $\bar{X}$ in the encoded space is equivalent to the Z gate on the unencoded qubit. The state $\bar{X}|\psi\rangle$ lies in the encoded or stabilized subspace. Any single qubit error X_i commutes with $\bar{X}$ giving $\bar{X}X_i|\psi\rangle = X_i\bar{X}|\psi\rangle$. The state is not in the encoded subspace, would be detected by the syndrome detection and appropriately corrected. Similarly $\bar{X}Z_i|\psi\rangle = -Z_i\bar{X}|\psi\rangle$ would be detected and corrected up to a global phase.

Traversal implementations

In the above section we showed **transversal** implementations of some single qubit gates. These are fault tolerant, in part because they are transversal, as shown in Figure 12. An error in a single bit or an error in the operation on a single bit would be corrected and would not propagate. However not all single qubit gates can be implemented transversally.

The Hadamard transformation H on a single qubit can be written as $H = \frac{1}{\sqrt{2}}(X+Z)$. However, for the Shor 9 bit code, this is not equivalent to $\bar{H} = H^{\otimes 9}$. I think the Hadamard H gate and the phase gate ($S = \mathrm{diag}(1,i) = P_{\frac{\pi}{2}}$) cannot be implemented for the Shor 9-bit

code in a simple transversal way.

Both H and S can be implemented transversally for Steane's 7-bit code, which might be why it is used as an example by Nielson & Chuang in their discussion of fault tolerant computation.

Apparently $P_{\frac{\pi}{4}}$ is also a problem for the Steane code and requires additional ancilla bits to implement in a fault tolerant manner.

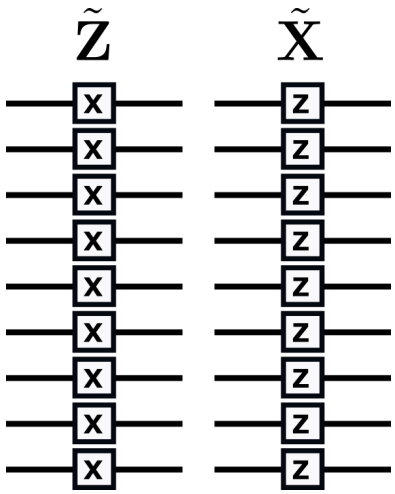

Figure 12: For Shor's 9-bit error correcting code, the single qubit Z, X operations can be implemented transversally in the 9 encoded qubits. This gives a fault-tolerant way of carrying out these operations.

7.3 Strategies for constructing fault tolerant operations

- Perform error correction steps periodically to ensure that errors don't build up.
- Perform error correction in a way that single errors do not propagate.
- Perform operations (gates) in such a way as to ensure that an error in any of the steps propagates to as few qubits as possible.
- Prepare states in such a way as to minimize errors in them.

Try to preform gates transversally.

Add in additional ancilla and steps to check the integrity of prepared states.

Partition calculations into pieces where only a single qubit is affected by each operation and each operation is separated by error correcting steps.

Use concatenated codes where there is a hierarchy of encoding.

7.4 Fault tolerant CNOT

The idea is to apply a CNOT gate in a transversal way. We can think of CNOT as $P_0 \otimes I + P_1 \otimes X$. Consider two encoded spaces for two qubits. We want to apply a series of operation. We would like each operation to depends on only a single qubit and affect only a single qubit in either control or target encoded spaces.

The simplest example is for the 3 bit bit flip error quantum code, as shown in Figure 13

$$|\psi\rangle = (a\,|0\rangle + b\,|1\rangle) \otimes (c\,|0\rangle + d\,|1\rangle)$$
$$CNOT\,|\psi\rangle = a\,|0\rangle \otimes (c\,|0\rangle + d\,|1\rangle) + b\,|1\rangle \otimes (d\,|0\rangle + c\,|1\rangle).$$

We now encode both bits

$$|\psi\rangle = (a\,|000\rangle + b\,|111\rangle) \otimes (c\,|000\rangle + d\,|111\rangle)$$

We apply CNOT separately to each pair of encoded bits

$$CNOT_{c=1}^{t=4} CNOT_{c=2}^{t=5} CNOT_{c=3}^{t=6}\,|\psi\rangle = a\,|000\rangle\,(c\,|000\rangle + d\,|111\rangle) + b\,|111\rangle \otimes (d\,|000\rangle + c\,|111\rangle)$$

If a single bit flip error occurs, then it would be correctly identified and corrected.

Examples of how to implement the fault tolerant $\pi/8$ and Toffoli gate are discussed by Nielson and Chuang in their book. A sufficient set of gates can be given in a fault tolerant manner to give the impression that it is possible to carry out a complex series of operations, even on a noisy quantum computer.

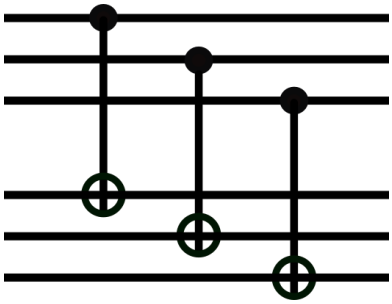

Figure 13: Implementation of a fault tolerant CNOT. The operation is performed with 3 separate operations that only affect a single bit in the encoding spaces. The top three bits are the encoding space for the control bit and the bottom three bits are the encoding space for the target bit, where each has been encoded using the 3-bit bit flip error correcting code.

8 Topological Error Correcting Codes

8.1 Surface Codes

Surface error codes are stabilizer codes operating on a 2-dimensional lattice of qubits where the stabilizers are composed of local operators (meaning operating on qubits that are near each other on the lattice). Both the logical qubits and the ancilla or data qubits are arranged on a lattice, as shown in Figures 14 – 16 .

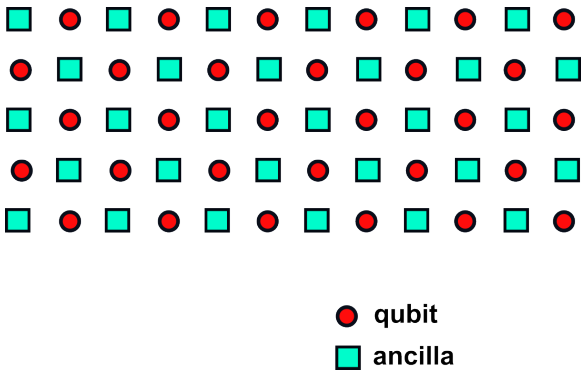

Figure 14: In a surface code, qubits, shown with red circles, are in a lattice. Ancilla qubits, shown with cyan squares, are also in a lattice.

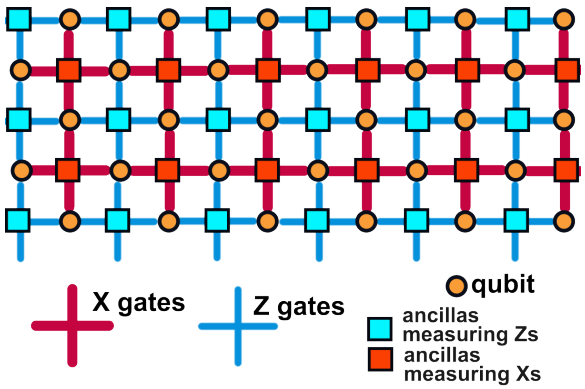

Figure 15: Ancilla qubits either measure the product of 4 Pauli X-operators or they measure the product of 4 Pauli Z operators. The ancillas that are connected by 4 thick red lines to 4 qubits measure the product of the 4 Pauli X operators associated with those 4 qubits. Similarly the ancillas that are connected by 4 thinner blue lines to 4 qubits measure the product of the 4 Pauli Z operators associated with those qubits.

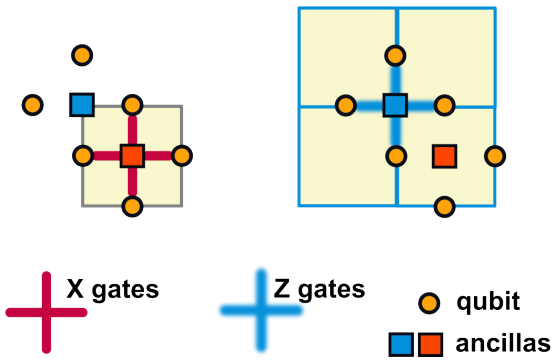

Figure 16: We can place each qubit on the edge of a square. The ancilla bit that is in the center of a square measures the product of X operators for the qubits on the square's edges (as shown on the left). The ancilla bit that is located on a square's vertex measures the product of Z operators for the qubits that are on edges that contain this vertex (as shown on the right).

8.2 A 5 bit surface code

In order of the labelled ancilla qubits, the generators for the 5 qubit surface code shown in Figure 17 are

$$\langle Z_1 Z_2 Z_3, X_1 X_3 X_4, X_2 X_3 X_5, Z_3 Z_4 Z_5 \rangle. \tag{33}$$

Do these operators commute with each other? The ancillas A_1 and A_2 give operators $Z_1 Z_2 Z_3, X_1 X_3 X_4$. The figure shows that they both involve operators on qubits 1 and 3. X_1 and Z_1 anticommute but Z_3 and X_3 also anticommute. Hence operators $Z_1 Z_2 Z_3, X_1 X_3 X_4$ commute. Each pair of neighboring ancillas involve operators on 2 shared qubits, so the operators generated from them commute.

What about ancillas A_1 and A_4? These give operators $Z_1 Z_2 Z_3, Z_3 Z_4 Z_5$ which commute. So the operators in the list of equation 8.2 all commute with one another. This is a generating set that generates a stabilizer group that lacks $-I$. Hence the list of operators in equation are a generator set for a stabilizer group.

What types of errors can this code correct? Apparently it can correct all single qubit errors so that it has distance $d = 2$ and it only encodes a single qubit worth of information.

The distance of a surface code can be increased by increasing the size of the lattice. With $d = \lambda$, apparently the generated code is $[[n = \lambda^2 + (\lambda - 1)^2, k = 1, d = \lambda]]$.

8.3 Loops

In what sense are these codes considered topological? We attempt to give a feeling for what is meant with Figure 18.

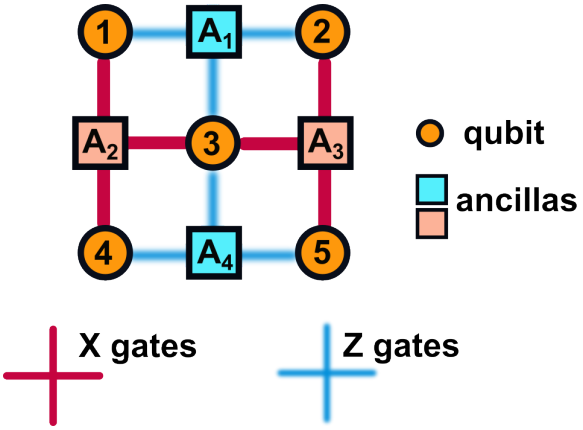

Figure 17: A 5 qubit surface code $[[5, 1, 2]]$. The A_1 ancilla is associated with the stabilizer generator $Z_1Z_2Z_3$. The A_2 ancilla is associated with the stabilizer generator $X_1X_3X_4$.

8.4 Toric Codes

A toric code is a surface code covering a rectangle, but with edges connected in such a way that the surface forms a torus.

8.5 Q-dit toric codes

Surface codes can be generalized to operate for qudits (which have more than 2 states). See [5]

The geometry of the error code is similar, with qudits spaced on the edges of a square lattice and ancilla qudits at either vertices or faces of each square. Instead of using the Pauli X, Z operators for ancilla measurements generalized operators $\hat{Z}, \hat{X}$, called clock and shift operators, are used to describe measurements.

$$\hat{X} = \sum_{j=0}^{n-1} |j+1\rangle \langle j| \qquad \hat{Z} = \sum_{j=0}^{n-1} \omega^j |j\rangle \langle j| . \tag{34}$$

where $\omega = e^{2\pi i/n}$ and n is the number of states in the qudit. The clock operator is the Fourier transform of the shift operator and vice versa. Even though these operators are not Hermitian, they can be effectively used to describe measurements in a particular basis and in its associated Fourier transform basis, respectively.

An ancilla measuring the product of 4 X operators in a conventional qubit surface code is replaced with an ancilla measuring in the basis of the product of 4 different $\hat{X}$ ops (one

[5] Fast decoders for qudit topological codes, Anwar, H. et al. 2014 New J. Phys. 16 063038. `https://iopscience.iop.org/article/10.1088/1367-2630/16/6/063038/meta`

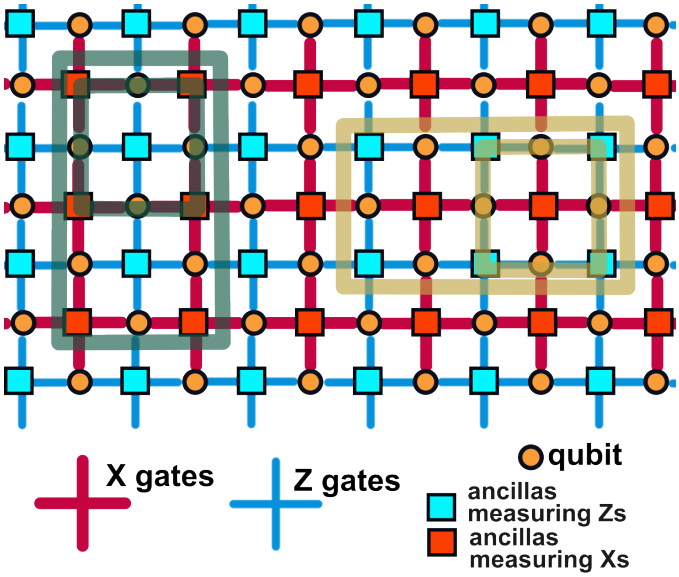

Figure 18: Consider the smaller green loop on the left. Suppose this symbolizes a chain of Z gates which is the product of 4 Z gates, where each one operates on the qubit that is covered by the loop. This loop also shows the stabilizer generator for the cyan square ancilla in the center of the loop. If we multiply this operator by the stabilizer generator of the ancilla directly below it, the result gives the larger green loop. Similarly on the right, the smaller orange loop shows a chain of X operators which is also equal to the stabilizer generator associated with the red ancilla inside it. If we multiply by the stabilizer elemetn associated with the red ancilla to the left, we find an operator symbolized by the larger orange loop that is associated with a larger chain of X operators. Members of the stabilizer group contain elements that are associated with loops.

for each qudit). Similarly an ancilla measuring the product of 4 Z operators (in the qubit code) is replaced with an ancilla measuring in the basis of the product of 4 different $\hat{Z}$ ops (one for each qudit).

9 Measuring errors on a quantum computer

Up to this point we have discussed correcting errors but not yet discussed characterizing the size of errors on real system. Supposing one can perform quantum operation but one does not know what the operation does. This can be done in detail with a method known as **quantum process tomography**.

9.1 Quantum process tomography

Any quantum channel, since it is linear, can be described in terms of a large matrix, known as the χ representation. The idea is to vectorize the density operator and describe how the channel operates on the vector. In a d dimensional quantum space, instead of considering the density matrix as a $d \times d$ matrix, turn it into a d^2 dimensional vector. The channel can be described by a $d^2 \times d^2$ matrix operating on the vectorized density operator and giving a new vectorized density operator. We can use the Hilbert-Schmidt inner product to create an orthonormal basis for the density operator. One possible basis is the set of all matrix elements $|i\rangle \langle j|$. However, it is not possible to *prepare* all of these states. Instead a basis that is related to Pauli matrices (for all pairs of states so $O(d^2)$) is used because density operators for these can be prepared. For example consider preparing the following set of density operators

$$\begin{aligned}
\frac{1}{2}|0\rangle\langle 0| + \frac{1}{2}|1\rangle\langle 1| &= \frac{1}{2}\begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix} = \frac{1}{2}I \\
|+\rangle\langle +| &= \frac{1}{2}\begin{pmatrix} 1 & 1 \\ 1 & 1 \end{pmatrix} = \frac{1}{2}(I + \sigma_x) \\
|-i\rangle\langle -i| = \frac{1}{\sqrt{2}}(|0\rangle - i|1\rangle)\frac{1}{\sqrt{2}}(\langle 0| + i\langle 1|) &= \frac{1}{2}\begin{pmatrix} 1 & -i \\ i & 1 \end{pmatrix} = \frac{1}{2}(I + \sigma_y) \\
|0\rangle\langle 0| &= \begin{pmatrix} 1 & 0 \\ 0 & 0 \end{pmatrix} = \frac{1}{2}(I + \sigma_z)
\end{aligned}$$

The first of these is an isotropic mixture, the remaining ones are constructed of pure states. For every pair of states $|i\rangle, |j\rangle$ we can prepare the above density matrices but substituting $|i\rangle$ for $|0\rangle$ and $|j\rangle$ for $|1\rangle$. Then for each of these prepared density operators, we operate with the channel $\mathcal{E}$. With the resulting density operator $\mathcal{E}(\rho)$, we then carry out the method known as quantum state tomography. This consists of measuring all Pauli-like operators for all pairs of states ($O(d^2)$). The resulting measurements can be used to slowly fill in the coefficients of the $d^2 \times d^2$ matrix for the χ representation of the channel. Needless

to say, quantum state tomography requires **a lot** of operations ($O(d^4)$). However, in many settings it is not necessary to characterize a channel exactly. For example, it may be enough to characterize an average error as we discuss in the next section.

9.2 Randomized benchmarking

Randomized benchmarking is a method for finding average errors for quantum operations. It is routinely used to characterize the quality of single and two-qubit gates.

Randomized benchmarking is robust against state preparation and measurement errors, and is more efficient than conventional quantum process tomography which requires many more measurements and operations.

Randomized benchmarking employs a small set of unitary operations (Clifford gates). For this set, we use assume:

- The noise of every gate in the set is independent of the other gates.
- The set of involved unitary operations is not universal (that's why Clifford gates are commonly used).
- The noise properties do not vary during the experiment.
- The noise is described using completely positive trace-preserving (CPTP) maps. (No memory effects).

The procedure:

1. Choose a random sequence of $m-1$ Clifford gates.
2. Apply the sequence to an initial state.
3. Find the inverse of the sequence and apply it.
4. Carry out a two-outcome POVM measurement to calculate the fidelity between the initial state and the output state. The two measurement outcomes are either that the state is the same as the initial state or it is not the same as the initial state. The fidelity is related to the probability of the measurement outcomes.

We define **fidelity** between two density operators

$$F(\rho,\sigma) = \left(\operatorname{tr}\sqrt{\sqrt{\rho}\sigma\sqrt{\rho}}\right)^2. \tag{35}$$

If the two operators are equal, $\rho = \sigma$, then the fidelity $F(\rho,\rho) = 1$ as density operators satisfy $\operatorname{tr}\rho = 1$.

If ρ is a **pure state** then we can write $\rho = |\psi\rangle\langle\psi|$, giving $\sqrt{\rho} = \rho$. The fidelity

$$\begin{aligned} F(\rho,\sigma) &= (\mathrm{tr}(\sqrt{\rho\sigma\rho}))^2 \\ &= (\mathrm{tr}\Big(\sqrt{|\psi\rangle\langle\psi|\,\sigma\,|\psi\rangle\langle\psi|}\Big))^2 \qquad \text{for } \rho \text{ pure state.} \end{aligned}$$

Since $\langle\psi|\,\sigma\,|\psi\rangle$ is a number, not a matrix,

$$\begin{aligned} F(\rho,\sigma) &= (\sqrt{\langle\psi|\,\sigma\,|\psi\rangle}\,\mathrm{tr}\,|\psi\rangle\langle\psi|)^2 \\ &= \langle\psi|\,\sigma\,|\psi\rangle = \mathrm{tr}(\rho\sigma) \qquad \text{for } \rho \text{ pure state.} \end{aligned}$$

If $\rho = |0\rangle\langle 0|$ then the fidelity is the same thing as the probability p_0 of the system described by density operator σ being measured in the $|0\rangle$ state,

$$F(|0\rangle\langle 0|\,,\sigma) = \mathrm{tr}(|0\rangle\langle 0|\,\sigma) = \langle 0|\,\sigma\,|0\rangle = p_0. \tag{36}$$

The set of steps in the random benchmarking procedure is repeated a number times and with different values for m, the number of gates used in the sequence. The result is a list of fidelities that depend on the number of applied gates, or m.

The following function is fit to the series of estimates for the fidelity:

$$F_{\mathrm{fit}}(m) = A_0 p^m + B_0. \tag{37}$$

The fit to the fidelity measurements gives you estimates for A_0, B_0 and most importantly p. Probably B_0 is dependent upon the final measurement error. And since an error in the initial state would propagate, we expect A_0 is sensitive to the error in the state preparation as well as the error coming from inaccuracy in each gate. Because the gates are applied m times, we expect the cumulative gate errors to scale with fitting parameter p. The average gate error (error per Clifford gate) is computed from p

$$\epsilon_{RB} = 1 - p - \frac{(1-p)}{2^n}. \tag{38}$$

Here n is the number of qubits in the system.

To roughly derive this expression we start by assuming that the operation of each gate gives noise that is described by the depolarization channel with

$$\rho' = \epsilon_{\mathrm{depolarization}}(\rho_i) = \alpha\rho_i + \frac{1-\alpha}{2^n}\mathbf{I}. \tag{39}$$

Here α is the probability that the state remains unchanged and $(1-\alpha)$ that a totally mixed state is returned. The initial density operator is ρ_i. The factor of 2^n arrises because the trace of the identity is equal to 2^n in an n-qubit system and the second term on the right side of in equation 39 should have a trace of 1.

If the depolarization channel is applied m times then the density operator becomes

$$\rho_m = \epsilon^m_{\text{depolarization}}(\rho_m) = \alpha^m \rho_i + \frac{1-\alpha^m}{2^n}\mathbf{I}. \tag{40}$$

Suppose we begin with the ground state, giving initial state $\rho_i = |0\rangle\langle 0|$. The probability that the ground state is measured from ρ_m is

$$\begin{aligned} p_0 &= \text{tr}(|0\rangle\langle 0|\,\rho_m) \\ &= \alpha^m + \frac{1-\alpha^m}{2^n} \\ &= \frac{2^n-1}{2^n}\alpha^m + \frac{1}{2^n} \\ &= a_0\alpha^m + b_0, \end{aligned} \tag{41}$$

with constants $a_0 = \frac{2^n-1}{2^n}$ and $b_0 = 2^{-n}$. It follows that the probability of measuring the same state as one started with decays exponentially with α. If the ground state is returned, the measured fidelity would be 1, as the initial state would be the same as the output state. Note that equation 41 is in the same form as equation 37 for the fidelity.

The actual fidelity measured will have different coefficients A_0, B_0 due to additional errors introduced in initialization and measurement. However the exponent measured $p = \alpha$ should depend on the depolarization channel parameter. The error introduced from each gate (ignoring initialization and measurement errors), only depends only upon α. Once you have measured α you can compute the error per application of each gate which would be

$$\begin{aligned} \epsilon &= 1 - \text{tr}\big(|0\rangle\langle 0|\,\rho'\big) \\ &= 1 - \alpha - \frac{1-\alpha}{2^n} \qquad \text{using equation 39.} \end{aligned}$$

Happily this is the same as equation 38 which we had previously just stated for the random benchmark error!

Our rough derivation shows that if the errors are described by the depolarization channel then equation 38 gives an estimate for the error per gate. However, we don't actually expect the noise in each gate to be described by the depolarization channel. However, if the gates are randomly chosen Clifford gates, then the noise of each gate behaves *on average* as if it was the depolarizing channel.

9.3 The Clifford group

What is the Clifford group?

On an n-qubit system, the Clifford group is generated by three gates, the Hadamard, S and CNOT gates. Here $S = \text{diag}(1, i) = \sqrt{Z}$. The gates H, S operate on any of the qubits. The CNOT gate operates on any pair of qubits.

The Clifford group Clif_n is defined as the group of unitary operators that normalize the generalized Pauli group for the n-qubit system $\mathcal{G}_n$;

$$\text{Clif}_n = \{V : VgV^\dagger \in \mathcal{G}_n \quad \forall g \in \mathcal{G}_n\}. \tag{42}$$

Recall that the generalized Pauli group $\mathcal{G}_n$ is generated by elements that are products of $1, -1, i, -i$ times a set of Pauli I, X, Y, Z operators which can operate on the different qubits (see equation 18). Elements of the Clifford group are called Clifford gates.

It is helpful to compute the following commutators for single qubit gates

$$HXH = Z, \quad HZH = X, \quad HYH = -Y$$

$$SXS = iX, \quad SZS = I, \quad SYS = iY.$$

This shows that $H_i, S_i \in \text{Clif}_n$ for i covering all the qubits. What does the CNOT do? When conjugating a single qubit gate on one qubit, it can produce a qubit gate operating on both qubits

$$\begin{aligned}
\text{CNOT}_{c=0}^{t=1} X_1 \text{CNOT}_{c=0}^{t=1} &= X_1 X_2 \\
\text{CNOT}_{c=0}^{t=1} X_2 \text{CNOT}_{c=0}^{t=1} &= X_2 \\
\text{CNOT}_{c=0}^{t=1} Z_1 \text{CNOT}_{c=0}^{t=1} &= Z_1 \\
\text{CNOT}_{c=0}^{t=1} Z_2 \text{CNOT}_{c=0}^{t=1} &= Z_1 Z_2.
\end{aligned}$$

Consider the procedure for random benchmarking. We assume a form for the noise induced by an operation. That means assume that applying a gate C actually applies both $\mathcal{E}$ and C where $\mathcal{E}$ is a quantum channel describing error or noise. Applying a sequence of m Clifford gates corresponds to the sequence of operations

$$\mathcal{E} \circ C_m \circ \mathcal{E} \circ C_{m-1} .... \mathcal{E} \circ C_2 \circ \mathcal{E} \circ C_1.$$

Consider a channel on a single qubit. We can expand the channel's Kraus operators in a sum $\sum_i \alpha_{i,0} I + \alpha_{i,x} X_i + \alpha_{i,y} Y_i + \alpha_{i,z} Z_i$ using Pauli operators on that qubit. The set $\{I/\sqrt{2}, X/\sqrt{2}, Y/\sqrt{2}, Z/\sqrt{2}\}$ are an orthogonal basis for 2×2 matrices via the Hilbert-Schmidt inner product ($\langle\langle A|B\rangle\rangle = \text{tr}\big(AB^\dagger\big)$) and operator norm. A similar basis, known as the Gellman basis can be used to expand a multiple qubit quantum channel. The resulting channel expansion is multiplied by a randomly chosen Clifford gate. Then you take the average over different randomly chosen Clifford gates. The resulting averaged channel should be independent of the polarization if the system is a single qubit. Likewise, the resulting averaged channel should be independent of any of the Pauli operators even if the system is larger than a single qubit. In other words this average should resemble the depolarization channel.

Taking an average over a finite group G (like the Clifford group) of a quantum operation $\mathcal{E}$ is called a *twirl*. The twirl of a channel $\mathcal{E}$ we denote $W_{\mathcal{E}}$ and it is the channel defined as

$$W_{\mathcal{E}} = \frac{1}{|G|} \sum_{U \in G} U \circ \mathcal{E}. \tag{43}$$

The Clifford group is nice because the twirl of a channel computed with the Clifford group is equal to the twirl of the channel averaged (via the Haar measure) over the full continuous group $U(2^n)$ (though I think the proofs of this are not necessarily obvious). That means you need only use operators randomly drawn from the discrete Clifford group (which is much simpler than $U(2^n)$) to compute the average integrated over the $U(2^n)$.

Let's be clearer about what is meant by equation 43. The twirl of a channel is also a channel (so it operates on a density operator)

$$W_{\mathcal{E}}(\rho) = \frac{1}{|G|} \sum_{U \in G} U^\dagger \mathcal{E}(U \rho U^\dagger) U \tag{44}$$

If G is a continuous group

$$W_{\mathcal{E}}(\rho) = \int_G U^\dagger \mathcal{E}(U \rho U^\dagger) U dU \tag{45}$$

where the integral is over the entire group. The measure here is called the Haar measure.

If $\mathcal{E}$ is described with a set of Kraus operators $\{M_i\}$ then

$$\mathcal{E}(\rho) = \sum_i M_i \rho M_i^\dagger \tag{46}$$

and

$$W_{\mathcal{E}}(\rho) = \frac{1}{|G|} \sum_{U \in G} \sum_i U^\dagger M_i U \rho U^\dagger M_i^\dagger U. \tag{47}$$

Let's create Kraus operators

$$K_i = \frac{1}{|G|} \sum_{U \in G} U^\dagger M_i U. \tag{48}$$

Then equation 47 becomes

$$W_{\mathcal{E}}(\rho) = \sum_i K_i \rho K_i^\dagger. \tag{49}$$

The twirl of a channel can be computed by averaging the individual Kraus operators over the group.

Random benchmarking relies on the fact that the twirl over the Clifford group is equivalent to the twirl computed over all unitary operations. The depolarization channel is the only type of channel that obeys the symmetry

$$U^\dagger \epsilon(U\rho U^\dagger)U = \epsilon(\rho)$$

for all $U \in U(n)$ and all density operators ρ.

A nice introduction to random benchmarking and with a good selection of references I found here:
`https://github.com/Qiskit/textbook/blob/main/notebooks/quantum-hardware/randomized-benchmar`
`ipynb`