Introduction to quantum computation: Quantum error correcting codes

Lecture Notes for Quantum Computing

Lecturer: Dorit Aharonov   Scribe: Zvika Brakerski

June 28, 2001

Summary of the seventh week of semester 2 (12.5.2001 - 17.5.2001).

Abstract

In previous lectures, we observed that there are problems which seem to be easy in quantum computation and become a lot harder in classical computation.

However, so far we assumed that our systems are completely noise-free which is untrue in realistic systems. As a matter of fact, realistic quantum systems are very noise-sensitive.

In this lecture, we examined the issue of error correction in quantum computation. While at first glance it appears that noise in a quantum channel can destroy the computation beyond repair, we have seen that there are ways to make the quantum computation more robust.

We discussed Shor's method for encoding each qubit using 9 qubits such that we can correct an error on one qubit. Then we went on to review a more sophisticated quantum error correction method — CSS codes.

1 Communication Scenario and Error Correcting Codes

Our communication scenario is one of a noisy channel. Alice on one side of the channel sends qubits to Bob on the other side. The channel might add noise to the transferred qubits (on the exact model of the noise, see section 4).

Alice and Bob each has a noise free quantum computer and can perform whichever computation they desire. Of course we are only interested in *efficient* computation performed by Alice and Bob.

Alice and Bob want to be able to be able to recover from possible noise in the channel. This will be done by somehow sending redundant information in such a way that even if an error occurs, Bob (the receiver of the information) can (with some high probability) recover the original message from Alice.

The form of adding redundant information we use is called *Quantum Error Correcting Codes* (QECCs) and is borrowed from classical information theory.

Definition 1 *A $[k, m]$-quantum code is a subspace of dimension k in the Hilbert space of m qubits.*

A word in the code is any vector in the subspace. We said we wanted to use this code to correct errors. Let us define that rigorously.

Definition 2 *A transformation R is said to be an error correcting transformation for a code C over some error space E if for all errors $\mathcal{E} \in E$, $R \circ \mathcal{E} \circ |c\rangle = |c\rangle$ for all codewords $c \in C$.*

Specifically we say that R corrects d errors over n qubits if for all error operators $\mathcal{E}$ that operate on d qubits only, $R \circ \mathcal{E} = I$.

In this lecture we discussed a very special case of error correcting procedures that matches the quantum case (in aspects of linearity and uniformity).

Definition 3 *A unitary transformation R is said to be a* quantum *error correcting transformation for a code C over some error space E if for all errors $\mathcal{E} \in E$:*
$R \circ (\mathcal{E} \circ |c\rangle \otimes |\vec{0}\rangle) = |c\rangle \otimes |\psi_{\mathcal{E}}\rangle$.

Meaning the transformation takes the errorous codeword into the corrected codeword and in addition writes on some excessive register some information that is a function of the error (and not of the original codeword).

Remark 1 *Note that according to linearity it is sufficient to find a unitary R that corrects errors on basis states in order to correct errors on all codewords.*

2 Obstacles in Achieving Quantum Error Correction

It may appear that the power of quantum computation is derived from the assumption that the computation is not noisy. One might think that since the coefficients of the quantum state are complex, using the quantum model enables handling variables of infinite precision and thus get additional computational power.

Noise in quantum computation take various forms:

Gate inaccuracies may cause errors in the computation. In real life, the implementation of a gate is often only an approximation of the desired operator. Therefore, errors are being inserted into the computation.

Even though we have shown that we can obtain a small deviation in the final result by building our circuit using accurate enough gates, this result requires precision of $\frac{\epsilon}{n}$ for a computation of size n. Therefore we cannot build gates in this form that would be accurate enough for a computation in any length.

Decoherence seems like a much more frightening type of errors. Even though we try to isolate the quantum system from the environment as much as we can, we cannot supply *total* isolation. Therefore, the interaction of the quantum system and the environment result in *decoherence* of the quantum state which is equivalent to a partial measurement of the state by the environment.

Such errors seem to be devastating as we cannot restore the original state after a measurement has occured.

Therefore, we would like to protect our computation from the effects of noise. In classical computation, we would do this by maintaining redundant information on the state that would enable reconstruction in case an error has occured. In quantum computation, however, this seems to be a harder task. This is due to a few reasons:

No cloning: It appears that the no-cloning theorem makes it impossible to implement repetition codes as it does not allow duplication of the state. Furthermore, an even more strict version of the theorem prevents duplicating "a significant part" of the information of the state.

In addition, even if we could duplicate the quantum state, recovering from an error would not be an easy thing to do since we'd need to measure the state and make some "majority argument".

Errors are continuous: Since the state is a vector of continuous elements, a series of continuous errors might occur. Finding out which error has occured in order to fix it, might require infinite precision.

Measurement destroys quantum information: As mentioned before, a measurement changes the state beyond recovery and therefore when we want to decode the state, we must not use measurements.

In spite of all of the above, quantum error correction is possible as we shall see.

3 Quantum Error Correction Sonnet

The ideas of this lecture are depicted very well in a sonnet written by Daniel Gottesman.[1]

Quantum Error Correction Sonnet

By Daniel Gottesman

We cannot clone, perforce; instead, we split
Coherence to protect it from that wrong
That would destroy our valued quantum bit
And make our computation take too long.

Correct a flip and phase - that will suffice.
If in our code another error's bred,
We simply measure it, then God plays dice,
Collapsing it to X or Y or Zed.

We start with noisy seven, nine, or five
And end with perfect one. To better spot
Those flaws we must avoid, we first must strive
To find which ones commute and which do not.

With group and eigenstate, we've learned to fix
Your quantum errors with our quantum tricks.

[1] The sonnet is published with author's permission.

4 Quantum Errors

A general quantum error in an n-qubit state can be represented by a linear transformation. We first consider the case of a single qubit and examine the possible errors and corresponding linear transformations.

The following four matrices represent a basis for all unitary one-qubit errors:

$$X = \begin{pmatrix} 0 & 1 \\ 1 & 0 \end{pmatrix}, Z = \begin{pmatrix} 1 & 0 \\ 0 & -1 \end{pmatrix}, XZ = \begin{pmatrix} 0 & -1 \\ 1 & 0 \end{pmatrix}, I = \begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix}$$

These matrices correspond respectively to bit-flip errors, phase-flip errors, phase and bit errors, and no errors, as can be seen by applying them to an arbitrary single-qubit state.

It is clear that the above matrices form a basis for all linear transformations on a single qubit. For n qubits, the basis matrices are the set of all n-item tensor products of the single-qubit matrices.

4.1 Noise Model

In a computation scenario, we divide the computation into discrete time steps. Each time step of the computation is followed by an application of a noise operator $\mathcal{E}$. Our model is a simplification of this model. Our model of sending a message over a noisy channel corresponds to a single time step of the computation. Nevertheless, in this section we also refer to the computation scenario.

Let us first consider unitary noise only (meaning no measurement is involved). In this case, the noise operator is in fact a tensor product of operators on each of the qubits:

$$\mathcal{E} = \mathcal{E}_1 \otimes \mathcal{E}_2 \otimes \cdots \otimes \mathcal{E}_n$$

Where $\mathcal{E}_i$ can be any unitary operator on one qubit. We see that our model of noise is *local* in space.

Now, what about decoherence? As we have seen in previous lectures, decoherence and measurements can be viewed as unitary operations on a larger space containing the state and the environment. Here we make two more locality assumptions:

1. We assume that our decoherence is local in space. Meaning that the environment of each qubit is *unique*. Therefore we can still write our noise operator as a tensor product of operators, each operating on a single qubit and a separate noise space.

2. We also assume that our decoherence is local in time. Meaning that in each time step, the environment is renewed. This assumption enables use of the noisy channel model.

Currently we make no assumptions as to the amount of noise we permit but as we shall see when we discuss actual codes, our codes will be efficient only if a small number of errors occurs (or the probability of error is small in probabilistic error models). This is very reasonable as it is obvious that if the noise is big enough, it can transform any state to any other state and destroy the information beyond repair (this also applies for classical computation). Furthermore, in real-life situation we build our systems to be as accurate and isolated as we can so it is reasonable to assume that we will be able to reduce the noise to levels we can handle.

4.2 Correcting Quantum Noise

We would like to be able to limit our discussions to the one-qubit operators discussed above. Therefore, we need to show that we do not lose generality by doing so.

Lemma 1 *It is sufficient to correct a basis for the error space in order to be able to correct all errors, including non unitary ones.*

Proof: It appears that the greatest obstacle in proving the lemma is taking care of the non-unitary errors, namely - decoherence. This is because it appears decoherence, unlike unitary operations, is irreversible.

Therefore, we first show that non-unitary errors do not make error correcting any harder.

Assume we operate on some n qubit state $|i\rangle$. We can assume that the initial environment is $|\vec{0}\rangle$.

A decoherence error would map this state as follows:

$$|i\rangle \otimes |\vec{0}\rangle \stackrel{\mathcal{E}}{\rightarrow} \sum_{j,e} |j\rangle \otimes |e\rangle$$

Such that $|j\rangle$ differs from $|i\rangle$ only in those qubits in which decoherence occured. Say there are d such qubits. There exists a unitary transformation that operates on those d bits only and takes $|i\rangle$ to $|j\rangle$. This transformation depends only on the error and not on i, j and therefore is corrected by our error correcting code.

Now, assume we have some error correcting transformation R which corrects d unitary errors on n qubits and is applied on the first n qubits of the final state, we get:

$$\begin{aligned}\sum_{j,e}(R\circ|j\rangle)\otimes|e\rangle &= \sum_{j,e}(|i\rangle\otimes|\psi_{\mathcal{E}}\rangle)\otimes|e\rangle \\ &= |i\rangle\otimes(|\psi_{\mathcal{E}}\rangle\otimes\sum_{e}|e\rangle)\end{aligned}$$

Therefore in the first n qubits we get our corrected code-word. Since we got a tensor product, tracing out the environment would not harm the first n qubits.

Since we assume that the environment is renewed in each time step, accumulation of errors would not hurt this result (we'd just have a longer environment vector).

Now that we figured out that decoherence does not effect error correcting, let's deal with unitary errors.

We want to show that we only need to correct a basis for the error space. To do this, we only need to show that error correction is linear. Once we have shown this, it is obvious that if we can correct a basis for the error space, we can correct all linear transformations of this basis, meaning the whole space.

Assume we have a code that corrects two kinds of errors: $\mathcal{E}_1$ and $\mathcal{E}_2$. Then we can correct any error which is a linear combination of $\mathcal{E}_1$, $\mathcal{E}_2$. This is because:

$$(c_1\mathcal{E}_1 + c_2\mathcal{E}_2)|\alpha\rangle = c_1\mathcal{E}_1|\alpha\rangle + c_2\mathcal{E}_2|\alpha\rangle$$

And then if we apply our error correcting transformation (say R):

$$\begin{aligned}R(c_1\mathcal{E}_1 + c_2\mathcal{E}_2)|\alpha\rangle &= Rc_1\mathcal{E}_1|\alpha\rangle + Rc_2\mathcal{E}_2|\alpha\rangle \\ &= c_1R\mathcal{E}_1|\alpha\rangle + c_2R\mathcal{E}_2|\alpha\rangle \\ &= c_1|\alpha\rangle + c_2|\alpha\rangle \\ &= |\alpha\rangle\end{aligned}$$

We corrected both errors.

Therefore, it is sufficient to correct a basis for the error space in order to be able to correct all errors spanned by that basis.

Furthermore, since we assume that errors are local is space, our error space is spanned by $\mathcal{B}^{\otimes n}$ where $\mathcal{B}$ is a basis for all *one-qubit* errors.

∎

5 Quantum Errors can be Corrected - Shor's Code

Shor has shown how to encode a one-qubit state $|\alpha\rangle$ into a 9-qubit state $|S_\alpha\rangle$ such that we can correct any one-qubit error.

We will see how to construct this.

5.1 Correcting Bit-Flips

Let's consider only bit-flips for the moment. We assume that the only thing that can happen to our codeword $|S_\alpha\rangle$ is having NOT operated on one of its qubits.

In classical computation, correcting such an error is very simple using repetition code. In quantum computation, we can use a similar concept. Consider the following coding:

$$C_b : |\alpha\rangle \otimes |\vec{0}\rangle \mapsto |S_\alpha\rangle$$

Such that:

$$\begin{array}{rcl} C_b(|0\rangle \otimes |\vec{0}\rangle) & \mapsto & |000\rangle \\ C_b(|1\rangle \otimes |\vec{0}\rangle) & \mapsto & |111\rangle \end{array}$$

It is very easy to show that if a bit flip occurs, this code fixes it. This is done by finding the majority of the values of the three qubits and writing the index of the bit that violates that majority (if such exists). We can do this in classical computation and there is no problem, therefore, in doing the same thing in quantum computation.

For example let's take an initial state:

$$|\alpha\rangle = c_0|0\rangle + c_1|1\rangle$$

After coding we get:

$$|S_\alpha\rangle = c_0|000\rangle + c_1|111\rangle$$

Now let's assume the channel flipped the second qubit:

$$|\tilde{S}_\alpha\rangle = c_0|010\rangle + c_1|101\rangle$$

Now we apply our error correction routine (which we call R):

$$\begin{aligned} R|\tilde{S}_\alpha\rangle &= c_0|010\rangle|2\rangle + c_1|101\rangle|2\rangle \\ &= (c_0|010\rangle + c_1|101\rangle) \otimes |2\rangle \end{aligned}$$

Then we measure the second register, obtain the index of the error and apply a NOT to correct the flip.

So we found a code that corrects bit flips. But we still need to handle phase flips and phase-flips followed by bit-flips.

5.2 Correcting Phase-Flips

All we have to do in order to correct phase flips, is realize that a phase flip is just a bit flip in the Hadamard basis.

This can be seen by presenting the bit-flip matrix in the Hadamard basis:

$$\begin{aligned} HXH^{-1} &= \begin{pmatrix} \frac{1}{\sqrt{2}} & \frac{1}{\sqrt{2}} \\ \frac{1}{\sqrt{2}} & -\frac{1}{\sqrt{2}} \end{pmatrix} \cdot \begin{pmatrix} 0 & 1 \\ 1 & 0 \end{pmatrix} \cdot \begin{pmatrix} \frac{1}{\sqrt{2}} & \frac{1}{\sqrt{2}} \\ \frac{1}{\sqrt{2}} & -\frac{1}{\sqrt{2}} \end{pmatrix} \\ &= \begin{pmatrix} 1 & 0 \\ 0 & -1 \end{pmatrix} \\ &= Z \end{aligned}$$

Therefore, we can use our bit-flip correcting transformation if we just encode our message in the Hadamard basis.

Explicitly we will encode against bit-flips as we did before, and then take our code word and present it in the Hadamard basis. A phase flip on the new code word is equivalent to a Hadamard basis representation of a bit-flip on the original code word. And therefore, can easily be fixed.

We will use:

$$C_p : |\alpha\rangle \otimes |\vec{0}\rangle \mapsto |S_\alpha\rangle$$

Such that:

$$\begin{aligned} C_p(|0\rangle \otimes |\vec{0}\rangle) &\mapsto |+++\rangle \\ C_p(|1\rangle \otimes |\vec{0}\rangle) &\mapsto |---\rangle \end{aligned}$$

Where:

$$\begin{aligned}|+\rangle &= \tfrac{|0\rangle+|1\rangle}{\sqrt{2}}\\ |-\rangle &= \tfrac{|0\rangle-|1\rangle}{\sqrt{2}}\end{aligned}$$

When we do this, a phase-flip can only change a $|\pm\rangle$ to a $|\mp\rangle$ and we can detect and correct this by applying Hadamard to all qubits and then using our error correction for bit-flips.

To make it clearer, let's do the math explicitly.

First of all, how does a phase-flip effect a state in the Hadamard basis?

$$\begin{aligned}|\alpha\rangle &= c_0|+\rangle + c_1|-\rangle\\ &= c_0\cdot\tfrac{|0\rangle+|1\rangle}{\sqrt{2}} + c_1\cdot\tfrac{|0\rangle-|1\rangle}{\sqrt{2}}\\ &= \tfrac{c_0+c_1}{\sqrt{2}}\cdot|0\rangle + \tfrac{c_0-c_1}{\sqrt{2}}\cdot|1\rangle\end{aligned}$$

And when we apply a phase-flip, we get:

$$\begin{aligned}|\alpha\rangle &\xrightarrow{Z} \tfrac{c_0+c_1}{\sqrt{2}}\cdot|0\rangle - \tfrac{c_0-c_1}{\sqrt{2}}\cdot|1\rangle\\ &= c_0\cdot\tfrac{|0\rangle-|1\rangle}{\sqrt{2}} + c_1\cdot\tfrac{|0\rangle+|1\rangle}{\sqrt{2}}\\ &= c_0|-\rangle + c_1|+\rangle\end{aligned}$$

And we see that in the Hadamard basis, the bit has been flipped.

Now let's see our encoding in action. Assume we start with some superposition:

$$\begin{aligned}|\alpha\rangle &= c_0|0\rangle + c_1|1\rangle\\ |S_\alpha\rangle &= c_0|+++\rangle + c_1|---\rangle\end{aligned}$$

Assume a phase-flip occured in the second qubit, we get:

$$|\tilde{S}_\alpha\rangle = c_0|+-+\rangle + c_1|-+-\rangle$$

In the decoding phase, we switch back to the standard basis (by applying $H^{\otimes^3}$) and get:

$$|\tilde{S}_\alpha\rangle = c_0|010\rangle + c_1|101\rangle$$

And then we can correct the bit-flip received using the procedure from the previous section.

5.3 Correcting Bit-Flips and Phase-Flips

We have a code that corrects bit-flips and a code that corrects phase flips. In order to correct both errors, we simply use one code on top of the other.

Specifically, our encoding is:

$$C_{bp} : |\alpha\rangle \otimes |\vec{0}\rangle \mapsto |S_\alpha\rangle$$

Such that:

$$\begin{array}{lcl} C_{bp}(|0\rangle \otimes |\vec{0}\rangle) & \mapsto & \frac{|000\rangle+|111\rangle}{\sqrt{2}} \otimes \frac{|000\rangle+|111\rangle}{\sqrt{2}} \otimes \frac{|000\rangle+|111\rangle}{\sqrt{2}} \\ C_{bp}(|1\rangle \otimes |\vec{0}\rangle) & \mapsto & \frac{|000\rangle-|111\rangle}{\sqrt{2}} \otimes \frac{|000\rangle-|111\rangle}{\sqrt{2}} \otimes \frac{|000\rangle-|111\rangle}{\sqrt{2}} \end{array}$$

A bit-flip on this superposition is just like a bit-flip in any other superposition, a bit that reverts its value. A phase-flip, however, changes the **sign** of one of the three-qubit bell-state, changing it from $\frac{|000\rangle\pm|111\rangle}{\sqrt{2}}$ to $\frac{|000\rangle\mp|111\rangle}{\sqrt{2}}$.

Therefore correcting the errors is very easy. First we apply bit-flip correction to all three triplets. Since phase-flips don't effect the value of the bits, this will correct the bit-flip if such occured. Since at this point all three qubits of all triplets are identical, we can take just one qubit from each triplet. This is used as input for the phase-flip correction procedure. We get the state encoded by C_p. All we have to do is use the decoding we have for C_p and get the encoded qubit.

Important note: In my analysis there is a hidden assumption that the phase-flip occurs prior to the bit-flip. However since $ZX = -XZ$, this does not hurt generality.

6 Good Codes

In real-life scenarios we cannot assume that a block of qubits can only be corrupted to some degree (say only one qubit can be damaged as we assumed in Shor's code).

From now on we will discuss noisy channels with bit corruption probabilities. Such a channel is characterized by a parameter η. For each qubit that is being transferred across the channel, the probability that the bit gets corrupted is at most η. The probability for corruption of a qubit is independent of the other qubits or any other parameter (memoryless channel).

On such channels, we would like to define *good* sets of codes that will enable encoding any message in such a way that we can recover the information with high probability.

Definition 4 *A set of* $[k_i, n_i]$*-error correcting codes is said to be* good *if:*

1. *The* code ratio $R_i = \frac{k_i}{n_i}$ *for all codes in the set is bounded away from* 0*. Meaning there exists a positive* R *such that* $\forall_i . \frac{k_i}{n_i} \geq R$*.*

2. *There exist positive* η *and* p *such that if the bit error rate is no more than* η*, the probability for recovering the encoded codeword is no less than* p*.*

Shor's code as we defined it is a $[1, 9]$ QECC. We can extend it into a set of $[k, 9k]$ QECCs very easily be encoding each qubit according to Shor's code.

We get a code of constant ratio of $\frac{1}{9}$. But assuming our noise parameter is η, we would require that in each 9-qubit block, at most one qubit is corrupted in order to fully recover from the error. Therefore, the probability of recovering is no more than $[(1-\eta)^8 \cdot (1+8\eta)]^k$ which is an exponentially decreasing probability that is not bounded away from 0.

Hence, we need to find other ways to generate good QECCs. CSS codes may present such a way.

7 Review of Linear Error Correcting Codes (ECC)

We consider an error correcting code $C \subseteq \mathbb{F}_2^n$ with $|C| = 2^k$. Such a code has two important matrices, the $k \times n$ generator matrix G and the $(n-k) \times n$ parity matrix P.

7.1 The Generator Matrix

The rows of the generator matrix G span the codewords of C so that a general message m can be encoded to codeword c via

$$mG = c.$$

Here, $m \in \mathbb{F}_2^k$ and $c \in \mathbb{F}_2^n$. In general, such a code is classified as an $[n, k, d]$ code where n and k are as above and d is defined as the minimum Hamming distance between any two codewords in C. We note that since C is a linear code (*i.e.* $c_1, c_2 \in C \Rightarrow c_1 + c_2 \in C$), d may be equivalently defined as the minimum Hamming weight among non-zero codewords. A general $[n, k, d]$ code allows for the detection of up to $d-1$ bit errors (these errors would necessarily produce an invalid codeword), and allows for the correction of up to $(d-1)/2$ errors (by choosing the codeword of minimum Hamming distance from the received message).

7.2 The Parity Matrix

The parity matrix P has the property that its columns generate $C^{\perp}$ and thus

$$cP = 0 \Longleftrightarrow c \in C.$$

For a general received message of the form $c' \equiv c + e$ where $c \in C$ and e represents errors in the received message, the syndrome of c' is defined to be

$$c'P = (c + e)P = eP.$$

For $0 \leq |e| \leq d - 1$, eP will always be non-zero and hence permit error detection. Moreover, if $|e| \leq (d-1)/2$, the error e can be determined exactly and hence corrected. To show this, we observe the following claim:

Claim 1 *Assume e_1, e_2 are errors such that: $0 < e_i \leq \frac{d-1}{2}$, then they have different syndromes: $e_1 P \neq e_2 P$.*

Proof: Assume $e_1 P = e_2 P$, then $(e_1 - e_2)P = 0$ and therefore $(e_1 - e_2) \in C$ and since $e_1 - e_2 \neq 0$ and $w(e_1 - e_2) \leq d - 1$ we get a contradiction. ■

Therefore, syndrome decoding corrects all errors up to hamming weight of $\frac{d-1}{2}$.

Finally, we note that there exist classical $[n, k, d]$ ECCs with $n = O(k)$ and $d = \Theta(n)$.

8 CSS Codes

We go on to show a family of Quantum Error Correcting Codes (QECCs) that rely on linear codes. These codes are called CSS codes, and they are due to Calderbank, Shor, and Stein.[2]

8.1 Quantum Errors on Superpositions

We consider quantum states in a group G (for concreteness, consider $\mathbb{Z}_2^n$). In particular, consider a state $|H\rangle$ which is a uniform superposition over some subgroup $H \subseteq G$. That is,

$$|H\rangle = \frac{1}{\sqrt{|H|}} \sum_{h \in H} |h\rangle$$

[2]This section relies a great deal on lecture notes by François Labelle and Mark Pilloff taken in Umesh Vasirani's course in the Fall semester of 2000.

Then a bit-flip error will map $|H\rangle$ to a uniform superposition over a coset of H, namely the state

$$|H+g\rangle = \frac{1}{\sqrt{|H|}} \sum_{h\in H} |h+g\rangle$$

where g indicates which bits have been flipped. For example, with $G = \mathbb{Z}_2^n$, the nonzero bits of g correspond to the flipped bits of $|H\rangle$.

Similarly a phase error will map $|H\rangle$ to a superposition over H with weights determined by some character function $\chi_{g'} : G \to \mathbb{C}$. For the case $G = \mathbb{Z}_2^n$, the phase error induced by $\chi_{g'}$ on $|H\rangle$ yields the state:

$$\chi_{g'}|H\rangle = \frac{1}{\sqrt{|H|}} \sum_{h\in H} (-1)^{g'\cdot h}|h\rangle$$

Thus an initial state $|H\rangle$ will, after arbitrary errors, be a superposition of states of the form:

$$\chi_{g'}|H+g\rangle = \frac{1}{\sqrt{|H|}} \sum_{h\in H} \chi_{g'}(h+g)|h+g\rangle$$

We recall that under the Fourier Transform, bit-flip and phase errors of the type defined above are interchanged (see figure 1), so that the Fourier Transform of the preceding state is given by:

$$\mathrm{FT}_G : \chi_{g'}|H+g\rangle \mapsto \chi_g|H^\perp + g'\rangle$$

Here $H^\perp \subseteq G^*$ is defined as $\{k \in G^* | \forall_{h\in H}.\ k(h) = 1\}$ where G^* is the dual group of G.

8.2 CSS Implementation

We begin with 2 classical ECCs, C_1 and C_2, described by $[n_1, k_1]$ and $[n_2, k_2]$ respectively. We additionally require that $C_1 \subseteq C_2$ and that $C_1^\perp$ and C_2 are each capable of correcting t errors (thus the minimum distance between codewords in each is at least $d = 2t+1$). We now define an $[n, k_2 - k_1, d]$ code as follows. The codewords will be superpositions over cosets of C_1 in C_2. Then there are $|C_2|/|C_1|$ codewords each of the form

$$|x + C_1\rangle \equiv \frac{1}{\sqrt{|C_1|}} \sum_{y\in C_1} |x+y\rangle$$

where x is a representative element in C_2.

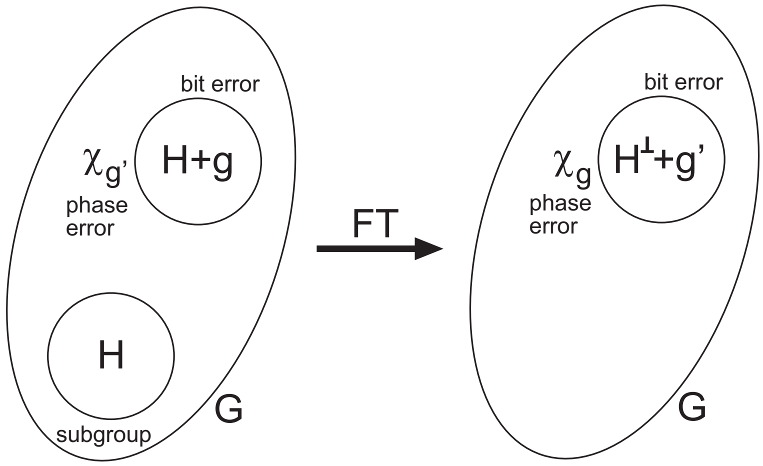

Figure 1: Quantum errors under the Fourier transform

Claim 2 *For all a, b, the corresponding codewords $|S_a\rangle = |a + C_1\rangle$ and $|S_b\rangle = |b + C_1\rangle$ are either identical or orthogonal.*

Proof: If $a - b \in C_1$ then obviously the two codewords are identical:

$$\begin{aligned}
|S_a\rangle &= \frac{1}{\sqrt{|C_1|}} \sum_{y \in C_1} |a + y\rangle \\
&= \frac{1}{\sqrt{|C_1|}} \sum_{y \in C_1} |b + (a - b) + y\rangle \\
&= \frac{1}{\sqrt{|C_1|}} \sum_{y' = (a-b)+y \in C_1} |b + y'\rangle \\
&= \frac{1}{\sqrt{|C_1|}} \sum_{y \in C_1} |b + y\rangle \\
&= |S_b\rangle
\end{aligned}$$

Otherwise, we get two different cosets of C_1 which are non-intersecting. Therefore $\langle S_a | S_b \rangle = 0$ since $|S_a\rangle$ and $|S_b\rangle$ are in orthogonal subspaces.

■

For concreteness we now consider the case $G = \mathbb{Z}_2^n$. Then as described in the previous section, arbitrary bit-flip and phase errors can be characterized by two group elements $e_b, e_p \in \mathbb{Z}_2^n$ respectively. The erroneous state is then

$$\frac{1}{\sqrt{|C_1|}} \sum_{y \in C_1} (-1)^{(x+y+e_b)\cdot e_p} |x + y + e_b\rangle \equiv \chi_{e_p} |C_1 + x + e_b\rangle.$$

The error correction algorithm proceeds in two stages. To correct the bit-flip errors, we first apply the parity matrix P_2 for the code C_2 and store the result in a set of ancillary qubits:

$$\chi_{e_p} |C_1 + x + e_b\rangle \otimes |0^n\rangle \xmapsto{P_2} \chi_{e_p} |C_1 + x + e_b\rangle \otimes |e_b P_2\rangle.$$

We now measure the second register and determine e_b as in the classical case. By applying the bit-flips indicated by e_b and discarding the ancillary qubits we arrive at the bit-flip corrected state

$$\chi_{e_p} |C_1 + x + e_b\rangle |e_b P_2\rangle \xmapsto{\text{correct}} \chi_{e_p} |C_1 + x\rangle.$$

To correct for phase errors, we apply a Fourier transform which, as described above, interchanges bit-flip and phase errors:

$$\chi_{e_p} |C_1 + x\rangle \xmapsto{\text{FT}} \chi_x |C_1^\perp + e_p\rangle.$$

From here we simply repeat the bit-error correction scheme above by applying the parity matrix $P_{1^\perp}$ for the code $C_1^\perp$ and correcting the measured error:

$$\chi_x |C_1^\perp + e_p\rangle |0^n\rangle \xmapsto{P_{1^\perp}} \chi_x |C_1^\perp + e_p\rangle |e_p P_{1^\perp}\rangle \xmapsto{\text{correct}} \chi_x |C_1^\perp\rangle.$$

By applying another Fourier transform we regain the original state:

$$\chi_x |C_1^\perp\rangle \xmapsto{\text{FT}} |C_1 + x\rangle.$$

9 On the Next Lecture

On the next lecture we will see that CSS codes over $\mathbb{F}_p$ can be defined. The proof for such codes works exactly the same as in this lecture.

We will see that polynomial codes give good CSS codes over $\mathbb{F}_p$.

Claim 3 (to be shown on the next lecture)

1. *We can construct polynomial codes that achieve rates of* $\frac{r+1}{m}$ *and correct* $\lfloor \frac{m-r-1}{2} \rfloor$ *errors.*

   This means that if we want to correct d *errors, we will pick* $m = (2d+1) \cdot (r+1)$ *and get a rate of* $\frac{1}{2d+1}$.

2. *A polynomial code of degree* d *with length* m *over* $\mathbb{F}_p$ *is a* $[1, m]$*-quantum code which corrects* $\min\{\lfloor \frac{m-d-1}{2} \rfloor, \lfloor \frac{d}{2} \rfloor\}$ *errors.*

   This code achieves a rate of $\frac{1}{m}$.