Error Correction Lecture 4

with Dan Browne

March 29, 2016

Contents

1 The Stabiliser Formalism

This is a state-independent formalism. It was first introduced by Gottesmann, This is by far one of the most important formalisms that made error correction a realistic prospect.

The Stabiliser formalism can correct any Pauli Error. It relies on the mathematical properties of Pauli matrices, and can also be used to explain multi-qubit entanglement. It is a very useful tehnique.

1.1 Example: 3-qubit repetition code

The 3-qubit repetition code is an example of the stabiliser formalism. For an arbitrary state

$$|\psi\rangle = \alpha|000\rangle + \beta|111\rangle \tag{1}$$

we can detect a X bit flip error by measuring ZZI, ZIZ or IZZ. Because the third measurement is superfluous, we need only measure the first two.

Recall the effect of an error on the qubits. We have

Error	States
XII	$\alpha\|100\rangle + \beta\|011\rangle$
IXI	$\alpha\|010\rangle + \beta\|101\rangle$
IIX	$\alpha\|001\rangle + \beta\|110\rangle$

Similarly, for a syndrome measurement using the previous measurements, we find

ZZI	ZIZ	IZZ
+	+	+
−	−	+
−	+	−
+	−	−

From this, we note that we are clearly dealing with a non-degenerate code. But instead of looking at the state and working out the entire syndrome table, we can ask: does the error commute with the error detection? That is,

$$[M, E] =^? 0 \tag{2}$$

We can summarise this in a table as well.

	ZZI	ZIZ	IZZ
III	√	√	√
XII	×	×	√
IXI	×	√	×
IIX	√	×	×

Note the similarity of the tables.

We can then prove that given an error E and error detection measurement M, if they anti-commute, we get -1 and if they commute, we get $+1$, should we measure.

So when M anti-commutes with the error E, we detect the error. Note that all the error detecting measurements commute. And note as well that on an error-free codewords, all outcomes of M are $+$ (which is exactly what we want) because

$$[M, I] = 0 \tag{3}$$

NOte talso that there is a product sturcture outcome of IZZ, since it is a product of outcomes ZIZ and ZZI. That is, we still only need to make two measurements because even the outcome of the third one will always be obtained from the other two.

These important group properties will come into play here. But before we proceed, we must show that the Pauli group is closed. THe identity, inverse and associativity are already taken care of.

2 Definition of the Stabiliser Group

We must first define the notion of **stabilisation**. A group element is stabilised if another group element leaves it invariant.

Thus, given a set of codewords space basis states, $|x\rangle_L$, where for $x = 00\ldots00$ to $x = 11\ldots11$ (which we can call binary vectors) we say that P stabilises the odespace if

$$P|x\rangle_L = |x\rangle_L \tag{4}$$

for all x. Note that this is an eigenvalue equation with eigenvalue $+1$. Thus, $|x\rangle_L$ is an eigenvector of P with eigenvalue $+1$ for all x.

In the Stabiliser formalism, we identify Pauli operators that we measure to detect errors with operators that stabilise our codewords. We saw the stabiliser for $|000\rangle$ and $|111\rangle$. They are ZZI, and ZIZ. What properties will they have?

It is not always true that anti-commuting operators do not share eigenvalues. Two Pauli operators share a joint eigenbasis if and only if they commute.

Consider therefore two commuting Pauli operators P and Q that both stabilise $|x\rangle_L$. We find that

$$P|x\rangle_L = |x\rangle_L \tag{5}$$

$$Q|x\rangle_L = |x\rangle_L \tag{6}$$

Then, it follows that

$$PQ|x\rangle_L = P|x\rangle_L = |x\rangle_L \tag{7}$$

So, PQ does also stabilise $|x\rangle_L$.

Thus, given a code space $|x\rangle_L$, its stabiliser group is the set of all Pauli operators S_j that satisfy

$$S_j |x\rangle_L = |x\rangle_L \tag{8}$$

for all x and for all j. The set $\{S_j\}$ forms a group! It is a subgroup of the N-qubit Pauli group $\{i, X, Z\}$.

Recall that a group with only commuting elements is abelian. The stabiliser group only has commuting operators, and hence is an abelian subgroup of the Pauli group.

Any quantum error correcting code for which the error detection measurements are an abelian subgroup of the N-qubit Pauli group is called a stabiliser group.

All codes seen in the course so far are in fact examples of stabiliser groups.

Exercise for the student: Go back and check this. Look at the commutators of the various operators.

For example, the 3-qubit code has stabilisers ZZI, ZIZ, IZZ, III. The 3-Qubit phase flip code has stabilisers XXI, XIX, IIX, III. Fruthermore, the Shor and the Steane code and the 5-qubit code are all examples of stabiliser codes. Finally, we have topological surface codes, which also fall into this category.

Note that nay group can be represented by its generators. Thus, for our future studies of stabiliser groups, we will have to look only at the stabiliser generators.

3 Order of group and group generators

The **order** of the group is the number of group elements.

Any group G can be described a set of generators (which is not unique) from which all elements can be obtained. We write this as

$$G = \langle g_1 \dots g_n \rangle \tag{9}$$

As an example, take the 3-qubit code. For the 3-qubit code, we have three possible sets of generators for the stabiliser group. $\langle ZZI, ZIZ \rangle$, $\langle ZIZ, IZZ \rangle$ and $\langle ZZI, IZZ \rangle$.

In general, a stabiliser group with m generators has 2^m elements. The generators must be **independent**. By independent, we mean that we choose the minimum number of generators that can generate the entire group. This is similar to choosing a vector basis.

For self-inverse elements, such as the Pauli group, 2^m also holds. Since $(g_j)^2 = I$ and since elements commute, the order of multiplication does not matter. Any element can in fact be written

$$S = \prod_{j=1}^{m} g_j^{x_j} \tag{10}$$

where x_j is an m-bit string. Then, we do indeed get 2^m elements.

So, m is always our generator index. In fact, describing the group by its generators gives us an exponential saving, since in order to speak of the group, we only have to consider m elements instead of 2^m elements.

However, how many generators do we need for the stabiliser formalism?

4 The Number of Generators

For the 3-qubit code, we had $m = 2$, $n = 3$ and $k = 1$. For the Shor code, we similarly had $n = 9$ and $k = 1$. The generators for that stabiliser group were $Z_1Z_2, Z_2Z_3, Z_3Z_4, Z_4Z_5, Z_5Z_6, Z_6Z_7, Z_7Z_8Z_8Z_9$ and $X_1X_2X_3X_4X_5X_6, X_1X_2X_3X_7X_8X_9$.

From these two examples, we can see a pattern. In general

$$m = n - k \tag{11}$$

This is satisfied by any stabiliser code. In words, given k qubits encoded in n physical qubits, the stabiliser has m independent generators. A proof can e found in one of the problem sheets.

The key idea is to use m generators to describe the entire code.

5 Aside:Stabiliser states

If $k = 0$ we end up with $m = n$. That is, we have no encoded states at all. We say that we have a 0-dimensional code space, which means that there is one single vector in the codespace. Note that they do not encode an entire qubit, since this requires two basis states. Instead, this code can be used to encode one single state.

States like these are called **stabiliser states**. It is a state which is the joint $+1$ eigenstate of n independent, commuting pauli operators (since $m = n$). A $k = 0$ code is also called a stabiliser code.

They are important because they use canonically entangled states.

Here are some examples of single states and their stabiliser generators.

State ($n = 1$)	Stabiliser generator
$\vert 0\rangle$	Z
$\vert 1\rangle$	$-Z$
$\vert +\rangle$	X
$\vert -\rangle$	$-X$
$\vert +i\rangle$	Y
$\vert -i\rangle$	$-Y$

And for states with more than one physical qubit,

States $n = 2$	Stabiliser generator	Full group
$\vert 00\rangle + \vert 11\rangle$	ZZ, XX	$ZZ, XX, -YY, II$
$\vert 00\rangle - \vert 11\rangle$	$ZZ, -XX$	$ZZ, -XX, YY, II$
$\vert 01\rangle + \vert 10\rangle$	$-ZZ, XX$	$-ZZ, XX, YY, II$
$\vert 01\rangle - \vert 10\rangle$	$-ZZ, -XX$	$-ZZ, -XX, -YY, II$

For $n = 3$ physical qubits, we end up with states such as the GHZ state: $|000\rangle + |111\rangle$ which is stabilised by $XXX; ZZI, ZIZ$. For any $n > 3$, we get so-called cluster states or graph states.

6 Detection of Errors in the Stabiliser Formalism

Consider a stabiliser codeword $|\psi\rangle$ and its stabiliser element S_j. Let this codeword be affected by a Pauli error E. The final state is $E|\psi\rangle$ and we measure S_j. Then, if

$$[S_j, E] = 0 \tag{12}$$

the outcome is $+1$. However, if

$$\{S_j, E\} = 0 \tag{13}$$

the outcome is -1 and we know that an error has occurred.

Proof: We want to know what the eigenstates of E applied to the state is. That is, what are the eigenstates λ_j in

$$S_j \left(E|\psi\rangle\right) = \lambda_j \left(E|\psi\rangle\right) \tag{14}$$

Assume first that $[S_j, E] = 0$. Then,

$$S_j E|\psi\rangle = E S_j|\psi\rangle = E|\psi\rangle \tag{15}$$

From which we see that the eigenvalue is indeed +1. However, if $\{S_j, E\} = 0$, we have

$$S_j E |\psi\rangle = -E S_j |\psi\rangle = -E |\psi\rangle \tag{16}$$

The eigenvalue is -1.

The central question in the stabiliser formalism is: does the error commute or anti-commute with the stabiliser?

7 Limitations of a Stabiliser Code

For a Pauli error E to be detectable, it must anti-commute with at least 1 element of the stabiliser. So, must we then go on to consider all elements in the group? It turns out that we can map the entire behaviour of the code by just considering the generators. If the error anti-commutes with just a single generator, it is detectable. This works because S_j is a product of generators.

So the only thing we need to measure are the generators. However, any error that commutes with all generators is undetectable. E.g. Z in the bit-flip code, which we already saw was undetectable. Recall that the generators were $[XXX, ZIZ, IZZ]$ and so a single Z error would indeed commute with them.

Question: It seems to me that Z does not commute with XXX. However, XXX is clearly a generator, as it maps the codeword $|000\rangle + |111\rangle$ into itself.

Attempt at answer: XXX only maps one single codeword onto itself, namely the superposition. For a state in $|000\rangle$, it does not map onto the same codeword. However, does this means that it is not a generator?

To clarify,

$$XXX\left(\alpha |000\rangle + \beta |111\rangle\right) = \alpha |111\rangle + \beta |000\rangle \tag{17}$$

Here, XXX is a logical X operator which maps codespace vectors back into the codespace. If an error is undetectable, it must be a logical operator on the codespace, and vice versa. In particular in stabiliser codes, undetectable Pauli errors are logical Pauli operators.

So, the logical Pauli operators are the set of logical operators that commute with the entire stabiliser.

Again, we can look at the bit flip code. The stabiliser itself is ZZi, IZZ, ZIZ, III. All of these are equal to the logical identity operator $\bar{I}$. These are <u>not</u> errors (but they could be, and then we couldn't possibly correct for them).

It turns out that every element of the stabiliser is always $\bar{I}$. But, note that for $XXX = \bar{X}$, we have

$$(XXX)S_j |\psi\rangle = (XXX) |\psi\rangle \tag{18}$$

So any product of the stabiliser and $\bar{X}$ is also a logical operator! So, given out three generators, we have four equivalent $\bar{X}$,

$$XXX, -YYX, -YXY, -XXY \tag{19}$$

The same goes for $\bar{Z}$ and $\bar{Y}$. This is a way to find all the undetectable errors. Recall that for the logical $\bar{Z}$ for the bit flip code, we have $\bar{Z} = ZII$. So we get

$$IZI, IIZ, ZZZ \tag{20}$$

are all equivalent $\bar{Z}$.

These are examples of **cosets**. In fact, we say that these are constructions of cosets. We can easily identify these sets of operators. This works for all stabiliser codes. We have that

$$\bar{Y} = \text{any logical } \bar{X} \times \text{any logical } \bar{Z} \tag{21}$$

We can prove this by writing

$$\bar{X}S_j\bar{Z}S_k = \bar{X}\bar{Z}(S_jS_k) \tag{22}$$

This creates four different logical $\bar{Y}$.

8 Overview of Error Correction

Let us here summarise what we have learnt so far.

Detection An error E is detected if it anti-commutes with at least one stabiliser generator.

Correction From the syndrome, we can determine a correction operator C. Then we apply C to the state.

Stabilisation It turns out that if C and E both form the stabiliser, such that

$$CE |\psi\rangle = S_j |\psi\rangle \tag{23}$$

then the error is successfully corrected. This way, we can easily find the right correction method. Otherwise, CE must be a logical operator. Note that error correction can always fail!

8.1 Successful correction example

Consider the error XII. If we apply the generators, we see that ZZI anti-commutes with the error and IZZ commutes. Thus, we get a -1 outcome for the first one and a $+1$ for the second one.

Then, we correct the error with XII. We see that

$$(XII)^2 = III \tag{24}$$

This is a stabiliser, and so the error is corrected.

Consider now the error IXX. It has syndrome -1 for ZZI and $+1$ for IZZ. We cannot correct this error because its syndrome is degenerate with the first case. Applying the same correction method will give us $XXX = \bar{X}$.

However, since 1 error is more likely than two, out chances are pretty good.

9 How to compute codeword kets

Let us now see how we recover the states from the Stabiliser formalism. That is, given a stabiliser formalism, what are the codewords? From the stabiliser generators and logical operators, we can find the states.

We first make an observation. Let $\langle g_1 \dots g_n \rangle$ be the stabiliser generators, and let $\bar{Z}$ be the logical Z operator. Then, $\left|0\right\rangle_L$ states obey

$$S_j \left|0\right\rangle_L = \left|0\right\rangle_L \tag{25}$$

$$\bar{Z} \left|0\right\rangle_L = \left|0\right\rangle_L \tag{26}$$

In fact $\left|0\right\rangle_L$ is the only state that satisfies both of these equations. However, this is cumbersome. Instead, we can create projectors onto this codespace. This leaves behind the codespacae vector.

So, for each g_j, we construct a projector

$$P_j = \frac{I + g_j}{2} \tag{27}$$

We can check that it is a projector,

$$P^2 = P = \left(\frac{I + g_j}{2}\right)^2 = \frac{I + g_j}{2} \tag{28}$$

Then, for any element in the codespace, we require

$$\left(\frac{i + g_j}{2}\right) \left|x\right\rangle = \left|x\right\rangle \tag{29}$$

Anything outside the codespace (such as an error) will give

$$g_j |\psi\rangle = - |\psi\rangle \tag{30}$$

We can rearrange the above to get

$$\left(\frac{I + g_j}{2}\right) |\psi\rangle = 0 \tag{31}$$

Form this we see that the projector indeed does only project onto the codespace. WE are essentially removing anything orthogonal to the codespace. We can take the product of all projectors,

$$\prod_{j=1}^{m} \left(\frac{I + g_j}{2}\right) = P \tag{32}$$

This is the projector onto the entire codespace. Every vector in the codespace is mapped to the codespace.

Now, let the projector be

$$\frac{I + \bar{Z}}{2} \tag{33}$$

It has the following properties. It leaves $|0\rangle_L$ unchanged, But, we find

$$\frac{I + \bar{Z}}{2} |1\rangle_L = 0 \tag{34}$$

Then we can show that

$$\prod_{j=1}^{m} \left(\frac{I + g_j}{2}\right) \left(\frac{I + \bar{Z}}{2}\right) = |0\rangle_L \langle 0|_L \tag{35}$$

because the $\bar{Z}$ projector picks out the same projector from P. Thus, we can easily obtain the codewords if we know the logical operators and the generators.

Another property is that

$$\prod_{j=1}^{m} \left(\frac{I + g_j}{2}\right) = \frac{1}{2^m} \sum_{\forall S \in \text{Stabiliser group}} S \tag{36}$$

10 5-qubit code

Let us now have a look at the 5-qubit code. It is defied within the Stabiliser formalism via its operators. It is a $(n = 5, k = 1, m = 4)$ code. We have

$$g_1 = XZZXI \tag{37}$$

$$g_2 = IXZZX \tag{38}$$

$$g_3 = XIXZZ \tag{39}$$

$$g_4 = ZXIXZ \tag{40}$$

The logical operators are (simply enough)

$$\bar{Z} = ZZZZZ \tag{41}$$

$$\bar{X} = XXXXX \tag{42}$$

From the stabilisers and the logical, we can work out the weight for the minimum undetectable error. We ask: What is the smallest weight logical operator?

We can calculate the distance of the code. Consider $\bar{Z}g_1$, which is another logical Z. We find

$$\bar{Z}g_1 = -YIIYZ \tag{43}$$

This is weight 3 (count all the operators that are not identity). This is the smallest weight undetectable error. Hence, the code distance is $d = 3$. The number of arbitrary errors that can be detected are

$$\frac{3-1}{2} \sim 1 \tag{44}$$

Note that this does <u>as well</u> as the Steane code. See NC for the entire list of the codewords, section 10.104. They are superpositions of 16 terms, so we will not write them out here!