Quantum Coding Theory **(UC Berkeley CS294, Spring 2024)**

Lecture 8: CSS Codes

February 16, 2024

Lecturer: John Wright *Scribe: Uma Girish*

Outline of this lecture

Last week we saw stabilizer codes and some necessary and sufficient conditions for the code to be non-trivial. It turns out that the conditions described in the previous lecture are not sufficient, so we will first correct them. Then, we will see two important examples of stabilizer codes.

- **The $[[5, 1, 3]]$ code [BDSW96, LMPZ96]:** This is the smallest code that encodes one logical qubit and corrects one error. We will see a description of this code.
- **Calderbank-Shor-Steane (CSS) codes [Ste96, CS96]:** This is a central family of quantum error correcting codes that are constructed from classical error correcting codes satisfying certain properties. We will discuss what these properties are and conclude with a useful description of the codewords in this code.

1 Stabilizer Codes

For a subgroup $S \subseteq \mathcal{P}_n$ of Pauli matrices we defined the stabilizer code $C[S]$ as the subspace of all vectors $|\psi\rangle \in \mathbb{C}^n$ that are stabilized by S, i.e., $g|\psi\rangle = |\psi\rangle$ for all $g \in S$. We think of the elements of S as the quantum analogue of *parity checks*. Under what conditions on the subgroup G is this subspace $C[S]$ non-trivial? Let us recall some necessary conditions. Let $g_1, \ldots, g_\ell$ be generators of the subgroup S, i.e., $S = \langle g_1, \ldots, g_\ell \rangle$. It is clear that these generators must satisfy the following conditions:

1. They commute, i.e., $g_i g_j = g_j g_i$ for all $i, j \in [\ell]$.
2. They are all observables, i.e., $g_i^2 = I$ for all $i \in [\ell]$.

Item 1 is necessary as any two Paulis g_i and g_j either commute or anti-commute, i.e., $g_i g_j = \pm g_j g_i$ and if they anti-commute, then there wouldn't exist any non-zero state $|\phi\rangle$ stabilized by both $g_i g_j$ and $g_j g_i$. Item 2 is necessary as the square of any Pauli matrix is $\pm I$, and for there to exist a non-zero $|\phi\rangle$ stabilized by this element, it would have to be I. While Items 1 and 2 are necessary, they are not sufficient. It turns out that we also need the following condition.

3. The negative identity is not in the subgroup, i.e., $-I \notin S$.

Clearly, we need the third condition as there is no state $|\psi\rangle$ that is stabilized by $-I$. Furthermore, this condition is not implied by the first two conditions. For instance, the Pauli subgroup $S = \{I, -I\}$, satisfies the first two conditions but not the third. We remark that Item 3 actually implies Item 2, since the square of every Pauli matrix is $\pm I$ and Item 3 rules out $-I$. It turns out that Item 1 and Item 3 are necessary *and also sufficient* conditions for the stabilizer code to be non-trivial. Let us describe this more formally now. A stabilizer code $C[S]$ is non-trivial if and only if the subgroup S satsifes the conditions below.

Definition 1.1 (Stabilizer code)**.** A stabilizer code $C[S]$ is defined by a Pauli subgroup $S = \langle g_1, \ldots, g_\ell \rangle \subseteq \mathcal{P}_n$ such that:

1. $g_i g_j = g_j g_i$ for all $i, j \in [\ell]$.
2. $-I \notin S$.

While the first condition here is easy to check (since it suffices to check that the generators commute), the second one is not super easy to check. A priori, there is no way to verify that $-I \notin S$ without checking every possible product of the generators. The reason is that there could exist some highly non-trivial combination of the generators that multiplies to the negative identity. For example, if we consider the subgroup $S = \langle XX, ZZ, YY \rangle$, we have that $(XX) \cdot (ZZ) = -YY \in S$ and since $YY \in S$, we would have $-(YY) \cdot (YY) = -I \in S$.

We remark that these conditions for the non-triviality of a quantum error correcting code seem to be a uniquely quantum phenomenon. In the classical world, any non-trivial subspace $S \subset \mathbb{F}_2^n$ of parity checks with dimension $\ell < n$ defines a classical code of non-zero dimension $n - \ell$.

1.1 Distance of a Stabilizer Code

Let us recall that formula for the distance of a stabilizer code. Firstly, what are the undetectable errors of a stabilizer code? These are precisely the elements of the centralizer $N(S)$ of S, where $N(S) = \{E \in \mathcal{P}_n : EP = PE, \forall P \in S\}$ consists of all Paulis that commute with the parity checks in S. If an error E satisfes $EP = PE$ for a parity check P, then E passes the parity check P and hence P cannot detect E. If this holds for all parity checks $P \in S$, then E becomes undetectable. We also saw that not all undetectable errors are important. For instance, if $E \in S$, then E is undetectable as it commutes with all the parity checks, however, it acts as the identity matrix on all the codewords, so effectively, it is as though no error was applied. What we really care about are undetectable errors that actually *change* the codewords. These are the precisely elements of $N(S) \setminus S$ - every element of this set is undetectable, and acts non-trivially on *some* codeword. Thus, we saw that the distance of the code is the minimum weight of an error $E \in N(S) \setminus S$.

$$\mathrm{dist}(C[S]) = \min_{E \in N(S) \setminus S} \mathrm{wt}(E). \tag{1}$$

Actually, there is a somewhat subtle point here that is often overlooked: if the Pauli P is in S, then $-P$ is in $N(S) \setminus S$ (as are $\pm i \cdot P$). But $-P$ is not an error per se, as for any state

$$\begin{array}{lccccc} g_1 = & X & Z & Z & X & I \\ g_2 = & I & X & Z & Z & X \\ g_3 = & X & I & X & Z & Z \\ g_4 = & Z & X & I & X & Z \\ \cancel{g_5 = & Z & Z & X & I & X} \end{array}$$

Figure 1: Generators of the parity checks for the $[[5, 3, 1]]$ code. The fifth generator here is the product of the first four and can be omitted.

$|\psi\rangle \in C[S]$, $-P$ just adds an undetectable global phase to $|\psi\rangle$. So we should not include $-P$ if we want to list out the undetectable errors. To fix this, we will slightly modify our definitions. Write $\mathcal{N}^+(S)$ for

$$\mathcal{N}^+(S) = \{P \in \{I, X, Y, Z\}^{\otimes n} \mid \forall Q \in S, PQ = QP\};$$

the "+" here denotes the fact that we are only considering the Paulis with a phase of $+1$. Similarly, write S^+ for

$$S^+ = \{P \in \{I, X, Y, Z\}^{\otimes n} \mid P \in S \text{ or } -P \in S\}.$$

Then the set of undetectable errors is *actually* $\mathcal{N}^+(S) \setminus S^+$. We can use this to revise our definition of the distance from Equation (1) to be

$$\text{dist}(C[S]) = \min_{E \in N^+(S) \setminus S^+} \text{wt}(E),$$

i.e. the smallest weight of any element in $\mathcal{N}^+(S) \setminus S^+$. In the literature, it is somewhat common to refer to $\mathcal{N}(S) \setminus S$ when $\mathcal{N}^+(S) \setminus S^+$ is meant; we will maintain this distinction throughout this lecture, although we will probably drop the distinction in future lectures.

We will now move on to discuss examples of stabilizer codes.

2 The $[[5, 1, 3]]$ Code

The $[[5, 1, 3]]$ code is a non-degenerate code on five physical qubits that encodes one logical qubit and corrects one error. This code was discovered independently by [LMPZ96] and [BDSW96]. The authors of [LMPZ96] arrived at this code by exhaustively searching for non-degenerate five-qubit codes that can correct one error. The $[[5, 3, 1]]$ code is also the smallest code that can correct single-qubit errors with one logical qubit (among all error-correcting codes, not necessarily non-degenerate) and this makes it interesting to experimentalists. We saw last week a code with one logical qubit that can *detect* one error using four physical qubits. But to *correct* one error, we need five qubits and the $[[5, 3, 1]]$ achieves this. This code is also known to be a *perfect code*, a code that saturates what is known as the quantum hamming bound which we will describe later. Let's define the $[[5, 3, 1]]$ code.

	X_1	X_2	X_3	X_4	X_5	Z_1	Z_2	Z_3	Z_4	Z_5	Y_1	Y_2	Y_3	Y_4	Y_5
g_1	+	-	-	+	+	-	+	+	-	+	-	-	-	-	+
g_2	+	+	-	-	+	+	-	+	+	-	+	-	-	-	-
g_3	+	+	+	-	-	-	+	-	+	+	-	+	-	-	-
g_4	-	+	+	+	-	+	-	+	-	+	-	-	+	-	-

Figure 2: Syndromes of single-qubit errors on the $[[5, 3, 1]]$ code. Each row represents a parity check and each column a single-qubit error.

Definition 2.1. The $[[5, 3, 1]]$ code is the Stabilizer code $C[S]$ on five qubits where

$$S = \langle XZZXI, IXZZX, XIXZZ, ZXIXZ, ZZXIX \rangle.$$

In other words, the parity checks of this code are generated by all possible cyclic shifts of $XZZXI$. This is depicted in Figure 1. It is not too difficult to see that the fifth generator is not needed as it is the product of the first four generators. It is not too hard to see that first four generators are independent: If a subset of the first four rows multiplies to the identity matrix, the first column tells us that this subset is contained in $\{1, 2, 3\}$, the second column tells us that it is contained in $\{2, 3, 4\}$ and the third column gives $\{1, 2, 4\}$ - the intersection of these is $\{2\}$ and this forces us to pick the empty subset.

2.1 Distance of the $[[5, 3, 1]]$ code

Let us see why this is a $[[5, 1, 3]]$ code. The number of physical qubits is 5 and the number of logical qubits is $5 - 4 = 1$ as we have four independent parity checks. We will now see that the distance of this code is 3, i.e., we correct single qubit errors but not more. Firstly, consider the error $X_1 := XIIII$ which acts as X on the first qubit and trivially on the rest. We can compute the syndrome of this error to be $|+, +, +, -\rangle$, and hence the fourth parity check allows us to detect this error. Similarly, $Z_1 = ZIIII$ has syndrome $|-, +, -, +\rangle$ and we can detect it using the first and third parity checks. Finally, $Y_1 = YIIII$ has syndrome $|-, +, -, -\rangle$ and we can detect it using the first, third and fourth parity checks. By symmetry, we can correct all single-qubit errors. This shows that the distance is at least 3 (since single-qubit errors can be corrected). It turns out that the distance is no bigger than 3. To see this, it is easy to construct weight-5 errors in $N^+(S) \setminus S^+$, namely $XXXXX$ and $ZZZZZ$ - these elements clearly commute with all of S and it is not too hard to see that these elements are not in S. To construct weight-3 errors in $N^+(S) \setminus S^+$, we observe that the error $IYYIX$ is equal to the product of a weight-5 error in $N^+(S) \setminus S^+$ (namely, $XXXXX$) and a parity check in S (namely, $XZZXI$), and hence we have a weight-3 error in $N^+(S) \setminus S^+$.

2.2 Non-degeneracy of the $[[5, 3, 1]]$ code

It turns out that the syndromes across all single-qubit errors are actually distinct, making the $[[5, 3, 1]]$ code a non-degenerate code. This means that we can actually identify which

error occurred using the syndrome. We can check this explicitly using the table in Figure 2. It turns out that this is the best you can do with a non-degenerate code of this size. In this code, there are 5 qubits and 3 possibilities for the error (X, Y or Z), so the number of possible errors we can correct is $5 \times 3 + 1$ ($+1$ for the identity error) which is 16. We only have 4 bits in our syndrome, so the number of possible syndromes is 16. Such kind of a code where the number of possible errors equals the number of possible syndromes is called a *perfect* code. In general, for a non-degenerate code, the number of possible errors must be at most the number of possible syndromes. This is essentially the content of the quantum hamming bound which we will see next week and the $[[5, 3, 1]]$ saturates the quantum hamming bound.

3 Calderbank-Shor-Steane (CSS) Codes

The CSS codes are a central family of codes. These were independently discovered by Calderbank and Shor [CS96] and by Steane [Ste96]. In the CSS code, each generator is a tensor product of (1) only X, I or (2) only Z, I. For example, $XXXII, XIXXI, ZIZII$ and $IIIIZ$ are examples of generators that satisfy this condition while $XZIII$ does not. We refer to generators of the form (1) as X parity checks and these come from some classical code. Similarly generators of the form (2) are called Z parity checks and come from another classical code. Provided these two classical codes satsify some nice properties, we obtain a good quantum error correcting code.

The motivation for these codes comes from the following. There are two sources of errors, namely, bit flips X and phase flips Z. It turns out that X parity checks are good at detecting Z errors and Z parity checks are good at detecting X errors. So the set of X parity checks effectively corrects for all Z errors while the Z parity checks correct X errors. Since $Y = XZ$, the hope is that correcting Z errors and X errors allows us to correct Y errors.

How do we choose the X parity checks and Z parity checks? The idea is to have both the X parity checks and Z parity checks come from good classical error correcting codes. The intuition is that a classical code is designed to correct bit-flip errors, namely X errors. Since the phase-flip errors Z act like bit-flip errors in the Z-basis, a classical code can also be used to correct Z errors using a change of basis. By combining the two, we obtain a code that can correct both X and Z errors. This idea allows us to import a lot of the intuition and ideas from the classical error correcting world into the quantum world. We now describe this in more detail.

3.1 Description of the CSS Code

First we set up some notation. Given $h \in \{0,1\}^n$, we use X^h to denote $X^{h_1} \otimes \ldots \otimes X^{h_n}$ where $X^a = X$ if $a = 1$ and I if $a = 0$. Let's recap classical linear codes. A $[n, k, d]$ classical linear code corresponds to an $\mathbb{F}_2$-linear subspace $C \subseteq \{0,1\}^n$ of dimension k. The code C is specified by the subspace of parity checks $C^\perp$, namely,

$$C = \{c : \langle c, h \rangle = 0 \bmod 2, \forall h \in C^\perp\}.$$

The distance of the code is the minimum hamming weight of a non-zero codeword, i.e.,

$$d := \min_{0 \neq c \in C} \mathrm{wt}(c).$$

Now that we have defined classical linear error correcting codes, we can define CSS codes. Let C_X, C_Z be classical linear codes where C_X is a $[n, k_X, d_X]$ linear code, and C_Z is a $[n, k_Z, d_Z]$ linear code. The idea here is to use C_X to detect phase-flip errors Z and use C_Z to detect bit-flip errors X. To do this, let us define the CSS code corresponding to C_X, C_Z as the stabilizer code with following two sets of generators:

- X^{h_X} for all $h_X \in C_X^\perp$,
- Z^{h_Z} for all $h_Z \in C_Z^\perp$

When does this work and produce a non-trivial stabilizer code? We need to check that the conditions in Definition 1.1 are satisfied. It is not too hard to see that $-I$ cannot be generated by multiplying $+I$, $+X$ and $+Z$ with each other. Next, we need to check the commutativity condition. In particular, we need that $X^{h_X} Z^{h_Z} = Z^{h_Z} X^{h_X}$ for all $h_X \in C_X^\perp, h_Z \in C_Z^\perp$. When does this hold? If $\langle h_X, h_Z \rangle = 1 \bmod 2$ then in the product $X^{h_X} Z^{h_Z}$ we get an odd number of Y and an overall phase of -1 whereas if $\langle h_X, h_Z \rangle = 0 \bmod 2$ then we get an even number of Y and an overall phase of 1. In particular,

$$X^{h_X} Z^{h_Z} = (-1)^{\langle h_X, h_Z \rangle \bmod 2} \cdot Z^{h_Z} X^{h_X}.$$

This implies that X^{h_X} and Z^{h_Z} commute iff $\langle h_X, h_Z \rangle = 0 \bmod 2$. In other words, every $h_X \in C_X^\perp$ is a parity check for every $h_Z \in C_Z^\perp$ and hence, $C_X^\perp \subseteq (C_Z^\perp)^\perp = C_Z$. Thus, the X parity checks are in the Z code and the Z parity checks are in the X code, i.e.,

$$C_X^\perp \subseteq C_Z \quad \text{and} \quad C_Z^\perp \subseteq C_X.$$

There's another way of writing this. Let $h_X^{(1)}, \ldots, h_X^{(\ell_X)}$ be a basis for $C_X^\perp$ and let $h_Z^{(1)}, \ldots, h_Z^{(\ell_Z)}$ be a basis for $C_Z^\perp$. Let us construct matrices whose rows consist of a basis for $C_X^\perp$ and $C_Z^\perp$ respectively, i.e.,

$$H_X := \begin{bmatrix} h_X^{(1)} \\ \ldots \\ h_X^{(\ell_X)} \end{bmatrix} \quad H_Z := \begin{bmatrix} h_Z^{(1)} \\ \ldots \\ h_Z^{(\ell_Z)} \end{bmatrix}.$$

The commutativity condition implies that

$$H_X \cdot H_Z^T = 0 \bmod 2.$$

This condition is what makes designing CSS codes so difficult and also so interesting. We need to design two codes, one that works for X parity checks and one that works for Z parity checks, such that both are both good codes, but also they have this nice relationship with each other, namely, the parity checks of each code are codewords of the other.

Definition 3.1. Let C_X, C_Z be classical linear codes where

$$C_X \text{ is a } [n, k_X, d_X] \text{ linear code, and}$$

$$C_Z \text{ is a } [n, k_Z, d_Z] \text{ linear code}$$

such that $C_X^\perp \subseteq C_Z$ and $C_Z^\perp \subseteq C_X$. The CSS code corresponding to C_X, C_Z is the stabilizer code with two sets of generators:

- X^{h_X} for all $h_X \in C_X^\perp$,
- Z^{h_Z} for all $h_Z \in C_Z^\perp$.

Let us now understand the parameters of the CSS code code. First, we have a code on n physical qubits. It is not too difficult to see that the generators $\{h_X^{(1)}, \ldots, h_X^{(\ell_X)}, h_Z^{(1)}, \ldots, h_Z^{(\ell_Z)}\}$ are independent group elements: The only way for a product of any subset of these generators to be I, is for the product within the X generators to be I and the product within the Z generators to be I, and since $h_X^{(1)}, \ldots, h_X^{(\ell_X)}$ forms a basis for $C_X^\perp$ and $h_Z^{(1)}, \ldots, h_Z^{(\ell_Z)}$ forms a basis for $C_Z^\perp$, the only way for this product to be I is to pick the empty subset. This gives us the following fact.

Fact 3.2. *Let $\ell_X = n - k_X$ and $\ell_Z = n - k_Z$. Then, CSS is a $[[n, n - \ell_X - \ell_Z]]$ code.*

3.2 Distance of the CSS code

Let's try to understand the distance of the CSS code. First, let's look at bit flip errors. Let $e \in \{0,1\}^n$. An X error on n qubits can be described by X^e where $X^e = X_1^{e_1} \otimes \ldots \otimes X_n^{e_n}$. To see the syndrome of this error, we need to see whether it commutes or anticommutes with out generators. Firstly, it commutes with all the X parity checks, i.e., $X^e X^{h_X} = X^{h_X} X^e$ for all $h_X \in C_X^\perp$. For a Z parity check $h_Z \in C_Z^\perp$ recall that

$$X^e Z^{h_Z} = Z^{h_Z} X^e (-1)^{\langle h_Z, e \rangle \bmod 2}.$$

Therefore, for each parity check $h_Z \in C_Z^\perp$, the syndrome of the error X^e corresponding to h_Z is $\langle h_Z, e \rangle \bmod 2$. Hence, the bit-flip error X^e is undetectable iff $\langle h_Z, e \rangle = 0 \bmod 2$ for all $h_Z \in C_Z^\perp$, in other words, $e \in C_Z$. Altogether, this tells us that for all $e \in \{0,1\}^n$,

$$\text{The error } X^e \text{ is in } N(S) \text{ iff } e \in C_Z.$$

This says that the undetectable X errors correspond to elements of C_Z. Suppose $0 \leq \mathrm{wt}(e) \leq d_Z - 1$, then by the distance property of code C_Z, we must have $e \notin C_Z$ and hence we can indeed detect the error X^e. This shows that we can detect X^e errors if $\mathrm{wt}(e) \leq d_Z$ and we can correct them if $\mathrm{wt}(e) \leq (d_Z - 1)/2$.

All of this shows that if a bit-flip error occurs, then we are able to correct it as long as the underlying classical code C_Z is able to correct those errors. But actually, we get something more, due to degeneracy. For example, take an undetectable error $e \in C_Z$ such that $e \in C_X^\perp$.

Since e is an X parity check, we have $X^e |\psi\rangle = |\psi\rangle$ for all states $|\psi\rangle$ in the CSS code. Thus, whenever $e \in C_X^\perp$, the error X^e acts trivially on all codewords and we have degeneracy. This means that the only undetectable errors that act non-trivially on the codewords, are those elements of C_Z that are *not in* $C_X^\perp$. In other words, the non-trivial bit-flip errors that can act on the code correspond to X^e where $e \in C_Z \setminus C_X^\perp$.

This motivates us to define the quantity

$$d_Z^+ := \min_{e \in C_Z \setminus C_X^\perp} \mathrm{wt}(e)$$

which correspond to the smallest non-trivial bit-flip errors that we cannot detect. This quantity is at least the distance d_Z of the classical code C_Z, since $d_Z := \min_{e \in C_Z} \mathrm{wt}(e)$ is a minimization problem over a larger set. And similarly

$$d_X^\perp := \min_{e \in C_X \setminus C_Z^\perp} \mathrm{wt}(e).$$

corresponds to the smallest non-trivial phase-flip error that we cannot detect. We can now show the following fact.

Fact 3.3. *Distance of the CSS code is precisely* $\min\{d_X^+, d_Z^+\}$.

Proof of Fact 3.3. Let us consider a general error E, which up to a global phase can be described by

$$E = X^{e_X} Z^{e_Z}$$

where $e_X, e_Z \in \{0,1\}^n$. We will show that the syndrome of this error under the various parity checks comprises of $\langle e_Z, h_X \rangle \bmod 2$ for each $h_X \in C_X^\perp$ and $\langle e_X, h_Z \rangle \bmod 2$ for each $H_Z \in C_Z^\perp$. This can be seen as follows.

- X parity checks: For any parity check $h_X \in C_X^\perp$, we have

  $$X^{h_X}(X^{e_X} Z^{e_Z}) = (X^{e_X} Z^{e_Z}) X^{h_X} \cdot (-1)^{\langle e_Z, h_X \rangle \bmod 2}.$$

  Thus, the syndrome corresponding to the parity check h_X is $\langle e_Z, h_X \rangle \bmod 2$.

- Z parity checks: For any parity check $h_Z \in C_Z^\perp$, we have

  $$Z^{h_Z}(X^{e_X} Z^{e_Z}) = (X^{e_X} Z^{e_Z}) Z^{h_Z} \cdot (-1)^{\langle e_X, h_Z \rangle \bmod 2}.$$

  Thus, the syndrome corresponding to the parity check h_Z is $\langle e_X, h_Z \rangle \bmod 2$.

Recall that the distance of the code is the minimum weight element of $N^+(S) \setminus S^+$. Let's first compute $N(S)$, the set of undetectable errors. By the above calculation, the error $E = X^{e_X} Z^{e_Z}$ is undetectable iff $\langle e_Z, h_X \rangle = 0 \bmod 2$ for all $h_X \in C_X^\perp$ and $\langle e_X, h_Z \rangle = 0 \bmod 2$ for all $h_Z \in C_Z^\perp$. This is equivalent to $e_Z \in C_X$ and $e_X \in C_Z$. Thus,

$$\text{The error } E = X^{e_X} Z^{e_Z} \text{ is in } N(S) \text{ iff } e_Z \in C_X \text{ and } e_X \in C_Z. \tag{2}$$

Now let's compute S, i.e., the errors which act as the identity matrix on the codewords. The error $E = X^{e_X} Z^{e_Z}$ is in S if and only if $e_Z \in C_Z^\perp$ and $e_X \in C_X^\perp$. Hence, we have the following condition.

$$\text{The error } E = X^{e_X} Z^{e_Z} \text{ is not in } S \text{ iff } e_Z \notin C_Z^\perp \text{ or } e_X \notin C_X^\perp. \tag{3}$$

Equations (2) and (3) imply that an error $X^{e_X} Z^{e_Z}$ is in $N(S) \setminus S$ if and only if $e_Z \in C_X$ and $e_X \in C_Z$ and either $e_Z \notin C_Z^\perp$ or $e_X \notin C_X^\perp$. In the first case, we obtain that $e_Z \in C_X \setminus C_Z^\perp$ and hence the weight of e is at least d_Z^+ and in the second case, we obtain that $e_X \in C_Z \setminus C_X^\perp$ and hence the weight of e is at least d_X^+. This shows that the distance is at least $\min(d_X^+, d_Z^+)$. It can be seen that this is also an upper bound by taking the error $X^{e_X} Z^{e_Z}$ that saturates this bound. □

As described earlier, we have $\min\{d_X^+, d_Z^+\} \geq \min\{d_X, d_Z\}$. In general, the distance of a CSS code can be strictly bigger than $\min(d_X, d_Z)$, due to degeneracy and we'll see the toric code in the future which is an example of this.

3.3 Describing the Codewords of the CSS code

The CSS code has the nice property that we can explicitly write down the codewords. For every $c_Z \in C_Z$, we can associate a codeword

$$\frac{1}{\sqrt{|C_X^\perp|}} \sum_{h_X \in C_X^\perp} |c_Z + h_X\rangle .$$

Observe that this codeword only depends on the subspace $c_Z + C_X^\perp$. In other words, we have a codeword for each element of the quotiented space $C_Z / C_X^\perp$. It's easy to see that it passes all Z parity checks, since $c_Z \in C_Z$ and $h_X \in C_X^\perp \subseteq C_Z$. Next class, we'll see that it passes X parity checks.

References

[BDSW96] Charles H. Bennett, David P. DiVincenzo, John A. Smolin, and William K. Wootters. Mixed-state entanglement and quantum error correction. *Phys. Rev. A*, 54:3824–3851, Nov 1996. (document), 2

[CS96] A. R. Calderbank and Peter W. Shor. Good quantum error-correcting codes exist. *Phys. Rev. A*, 54:1098–1105, Aug 1996. (document), 3

[LMPZ96] Raymond Laflamme, Cesar Miquel, Juan Pablo Paz, and Wojciech Hubert Zurek. Perfect quantum error correcting code. *Phys. Rev. Lett.*, 77:198–201, Jul 1996. (document), 2

[Ste96] Andrew Steane. Multiple-Particle Interference and Quantum Error Correction. *Proceedings of the Royal Society of London Series A*, 452(1954):2551–2577, November 1996. (document), 3