Last time we talked about the quantum Hamming code. This was based on the classical Hamming code. We have two matrices, $G$ and $H$, with

$$G = \begin{pmatrix} 0 & 0 & 0 & 1 & 1 & 1 & 1 \\ 0 & 1 & 1 & 0 & 0 & 1 & 1 \\ 1 & 0 & 1 & 0 & 1 & 0 & 1 \\ 1 & 1 & 1 & 1 & 1 & 1 & 1 \end{pmatrix} \quad \text{and} \quad H = \begin{pmatrix} 0 & 0 & 0 & 1 & 1 & 1 & 1 \\ 0 & 1 & 1 & 0 & 0 & 1 & 1 \\ 1 & 0 & 1 & 0 & 1 & 0 & 1 \\ 1 & 0 & 1 & 0 & 1 & 0 & 1 \end{pmatrix}$$

The quantum error correcting code they generated mapped $|0\rangle$ to

$$|0\rangle_L = \frac{1}{\sqrt{8}} \sum_{x \in H} |x\rangle,$$

and mapped $|1\rangle$ to

$$|1\rangle_L = \frac{1}{\sqrt{8}} \sum_{x \in H} |x \oplus 1111111\rangle.$$
Why are two cosets either equal or disjoint? Suppose that $y \in x + C_2$. Then $y = x + d$ for some $d \in C_2$. Now, if we add $d$ to any other element in $C_2$, we still have an element in $C_2$, so

$$y + C_2 = \{ z | z = y + c, c \in C_2 \} = \{ z | z = x + d + c, c \in C_2 \} = \{ z | z = x + c, c \in C_2 \} = x + C_2.$$ 

For example, in the Hamming code, there are two cosets of $H$ in $G$,

$$H \text{ and } H + 1111111.$$ 

Now, we can give the codewords of the CSS code associated with two $n$-bit classical codes $C_1$ and $C_2$ with $C_2 \subseteq C_1$. The codewords of this code will be the associated with the cosets of $C_2$ in $C_1$. For each coset $x + C_2$, we have the codeword

$$|x + C_2\rangle = \frac{1}{|C_2|^{1/2}} \sum_{c \in C_2} |x + c\rangle.$$ 

The number of codewords is the number of cosets of $C_2$ in $C_1$, which is $|C_2| / |C_2|$. The dimension of the code subspace is

$$\dim \text{CSS}(C_1 : C_2) = \log_2 \frac{|C_1|}{|C_2|} = \dim C_1 - \dim C_2.$$ 

Why does CSS($C_1, C_2$) correct $t_1$ bit-flip errors? We know that there is a classical error-correction procedure that will correct $t_1$ or fewer errors in any codeword $c_1$ of $C_1$. Now, the codewords of CSS($C_1, C_2$) are composed of superposition of quantum states $|x + c\rangle$, each of which is a codeword of $C_1$. Thus, applying this classical error-correction procedure in quantum superposition will correct up to $t_1$ bit-flip errors in the quantum code.

We now need to explain why the dual code will correct $t_2$ phase errors. Recall that the dual of a subspace $W$ of a vector space $V$ was

$$W^\perp = \{ x \in V | x \cdot w = 0 \ \forall w \in W \}.$$ 

If you’re used to working with real and complex vector spaces, a non-intuitive fact about binary vector spaces is that the original space and its dual can overlap. For example, with the Hamming code above, $H = G^\perp \subseteq G$. However, it is still true that if $W^\perp$ is the dual of $W$ in vector space $V$, then

$$\dim V = \dim W + \dim W^\perp.$$ 

We won’t prove it in these notes, but there is a proof of this along the lines of the Gaussian elimination manipulations we did in our discussion of Simon’s algorithm.

Let’s apply the Hadamard transform to the codewords $|x + C_2\rangle$. It turns out that what we get are superpositions of the codewords of the code CSS($C_2^\perp : C_1^\perp$). Before we prove this, let me make a little side comment about why this makes sense. because $C_2 \subseteq C_1$, any vector that is perpendicular to everything in $C_1$ is also perpendicular
to everything in $C_2$, so $C_1^\perp \subseteq C_2^\perp$. Further, $\dim C_1^\perp = n - \dim C_1$ and $\dim C_2^\perp = n - \dim C_2$, so
$$\dim \text{CSS}(C_1 : C_2) = \dim \text{CSS}(C_2^\perp : C_1^\perp).$$

Now, let’s show that the Hadamard transform of a codeword $|x + C_2\rangle$ of $\text{CSS}(C_1 : C_2)$ is a superposition of codewords of $\text{CSS}(C_2^\perp : C_1^\perp)$. Taking the Hadamard transform, we find:

$$H^{\otimes n} |x + C_2\rangle = \frac{1}{\sqrt{\dim C_2^\perp}} \sum_{c_2 \in C_2^\perp} |c_2 + x\rangle$$

$$= \frac{1}{\sqrt{\dim C_2^\perp}} \sqrt{\frac{2^n}{2^m}} \sum_{y \in Z_2^n} \sum_{c_2 \in C_2^\perp} (-1)^{\langle c_2 + x, y \rangle} |y\rangle$$

$$= \frac{1}{\sqrt{\dim C_2^\perp}} \sqrt{\frac{2^n}{2^m}} \sum_{y \in Z_2^n} (-1)^{\langle x, y \rangle} \sum_{c_2 \in C_2^\perp} (-1)^{\langle c_2, y \rangle} |y\rangle$$

Now I claim that
$$\sum_{c_2 \in C_2^\perp} (-1)^{\langle c_2, y \rangle} = \begin{cases} |C_2| & y \in C_2^\perp \\ 0 & y \notin C_2^\perp \end{cases}.$$ 

Why? if $y \in C_2^\perp$, all the elements of the sum are 1, so we get $|C_2|$. If not, then there is some $d \in C_2$ such that $(-1)^{\langle d, y \rangle} = -1$, and we can pair the elements of $C_2$ into pairs $(c_2, c_2 + d)$. Since $(-1)^{\langle c_2 + d, y \rangle} = 0$, the sum for each of these pairs is 0, so the whole sum is 0. Thus, we have

$$H^{\otimes n} |x + C_2\rangle = \frac{1}{\sqrt{\dim C_2^\perp}} \sqrt{\frac{2^n}{2^m}} \sum_{y \in C_2^\perp} (-1)^{\langle x, y \rangle} |C_2| |y\rangle$$

$$= \frac{\sqrt{\dim C_2^\perp}}{\sqrt{2^m}} \sum_{y \in C_2^\perp} (-1)^{\langle x, y \rangle} |y\rangle$$

$$= \frac{1}{\sqrt{\dim C_2^\perp}} \sum_{y \in C_2^\perp} (-1)^{\langle x, y \rangle} |y\rangle$$

Now, we are nearly done. We claim that if $y_1$ and $y_2$ are in the same coset of $C_1^\perp$ in $C_2^\perp$, then $(-1)^{\langle x, y_1 \rangle} = (-1)^{\langle x, y_2 \rangle}$. Why is this? Because $y_1 - y_2 \in C_1^\perp$ and $x \in C_1$, so $x \cdot (y_1 - y_2) = 0$. So now let $R$ be a set of representatives of the cosets of $C_1^\perp$ in $C_2^\perp$; this means there is one element of each coset in $R$. We can group the sum over $y$ above into elements from each coset of $C_1^\perp$ in $C_2^\perp$. This gives

$$H^{\otimes n} |x + C_2\rangle = \frac{1}{\sqrt{\dim C_2^\perp}} \sum_{y \in R} (-1)^{\langle x, y \rangle} \sum_{z \in y + C_1^\perp} |z\rangle$$

$$= \frac{\sqrt{\dim C_2^\perp}}{\sqrt{2^m}} \sum_{y \in R} (-1)^{\langle x, y \rangle} |y + C_1^\perp\rangle,$$

showing that the Hadamard transform of a codeword in $\text{CSS}(C_1 : C_2)$ is a superposition of codewords in $\text{CSS}(C_2^\perp : C_1^\perp)$, as we wanted.
The last thing I want to talk about in these notes is shifting a code. We will need this definition for the proof of the security of BB84, which we present in the next lecture. We can take a CSS code and shift it in bit space or in phase space. A codeword for a code shifted by \( s \) in bit space is

\[
|x + C_2\rangle = \frac{1}{|C_2|^{1/2}} \sum_{c \in C_2} |s + x + c\rangle
\]

and one shifted by \( t \) in phase space is

\[
|x + C_2\rangle = \frac{1}{|C_2|^{1/2}} \sum_{c \in C_2} (-1)^{t \cdot (x+c)} |x + c\rangle
\]

And of course, you can shift in phase space and bit space (the order makes a difference) to get

\[
|x + C_2\rangle = \frac{1}{|C_2|^{1/2}} \sum_{c \in C_2} (-1)^{t \cdot (x+c)} |s + x + c\rangle
\]

All of these shifted error correcting codes can correct the same number of errors as the unshifted ones. What we will need to know for the proof of security of quantum key distribution is that if you shift in both bit space and phase space by a random vector, and average over the resulting codes, you get the completely uniform distribution — the density matrix is \( I_{2^n} \). I’m not going to prove that in these lecture notes, but if you want to prove it, it would make a good exercise to make sure you understand CSS codes well.