Basics of Quantum Error Correction

Part 2

Stabilizers and CSS codes

John Watrous
IBM

Pauli operations

$$\mathbb{1} = \begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix} \qquad X = \begin{pmatrix} 0 & 1 \\ 1 & 0 \end{pmatrix} \qquad Y = \begin{pmatrix} 0 & -i \\ i & 0 \end{pmatrix} \qquad Z = \begin{pmatrix} 1 & 0 \\ 0 & -1 \end{pmatrix}$$

Anti-commutation relations:

$$XY = -YX \qquad XZ = -ZX \qquad YZ = -ZY$$

Multiplication rules:

$$XY = iZ \qquad YZ = iX \qquad ZX = iY \qquad XX = YY = ZZ = \mathbb{1}$$

An *n-qubit Pauli operation* is the n-fold tensor product of Pauli matrices. Its *weight* is the number of non-identity Pauli matrices in the tensor product.

$$\begin{array}{ll} \mathbb{1} \otimes \mathbb{1} \otimes \mathbb{1} \otimes \mathbb{1} \otimes \mathbb{1} \otimes \mathbb{1} \otimes \mathbb{1} \otimes \mathbb{1} \otimes \mathbb{1} & \leftarrow \text{weight } 0 \\ X \otimes X \otimes \mathbb{1} \otimes \mathbb{1} \otimes \mathbb{1} \otimes \mathbb{1} \otimes \mathbb{1} \otimes \mathbb{1} \otimes \mathbb{1} & \leftarrow \text{weight } 2 \\ X \otimes Y \otimes Z \otimes \mathbb{1} \otimes \mathbb{1} \otimes \mathbb{1} \otimes X \otimes Y \otimes Z & \leftarrow \text{weight } 6 \end{array}$$

Pauli operations as generators

$$\mathbb{1} = \begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix} \qquad X = \begin{pmatrix} 0 & 1 \\ 1 & 0 \end{pmatrix} \qquad Y = \begin{pmatrix} 0 & -i \\ i & 0 \end{pmatrix} \qquad Z = \begin{pmatrix} 1 & 0 \\ 0 & -1 \end{pmatrix}$$

Suppose that $P_1, \ldots, P_r$ are n-qubit Pauli operations.

The set *generated* by $P_1, \ldots, P_r$ includes all matrices that can be obtained from $P_1, \ldots, P_r$ by multiplication (taking any number of each operation and in any order).

Notation: $\langle P_1, \ldots, P_r \rangle$

Example 1

$$\langle X, Y, Z \rangle = \{\alpha P \,:\, \alpha \in \{1, i, -1, -i\},\ P \in \{\mathbb{1}, X, Y, Z\}\}$$ (16 elements)

Example 2

$$\langle X, Z \rangle = \{\mathbb{1},\ X,\ Z,\ XZ,\ -\mathbb{1},\ -X,\ -Z,\ -XZ\}$$ (8 elements)

Pauli operations as generators

$$\mathbb{1} = \begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix} \qquad X = \begin{pmatrix} 0 & 1 \\ 1 & 0 \end{pmatrix} \qquad Y = \begin{pmatrix} 0 & -i \\ i & 0 \end{pmatrix} \qquad Z = \begin{pmatrix} 1 & 0 \\ 0 & -1 \end{pmatrix}$$

Suppose that $P_1, \ldots, P_r$ are n-qubit Pauli operations.

The set *generated* by $P_1, \ldots, P_r$ includes all matrices that can be obtained from $P_1, \ldots, P_r$ by multiplication (taking any number of each operation and in any order).

Notation: $\langle P_1, \ldots, P_r \rangle$

Example 2

$$\langle X, Z \rangle = \{\mathbb{1}, X, Z, XZ, -\mathbb{1}, -X, -Z, -XZ\} \qquad \text{(8 elements)}$$

Example 3

$$\langle X \otimes X, Z \otimes Z \rangle = \{\mathbb{1} \otimes \mathbb{1}, X \otimes X, -Y \otimes Y, Z \otimes Z\} \qquad \text{(4 elements)}$$

Pauli observables

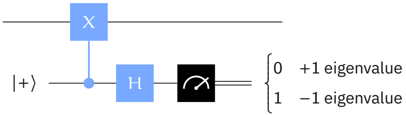
Pauli matrices describe unitary operations — but they also describe *measurements.*

More precisely, we can associate each Pauli matrix with a *projective measurement* defined by its eigenvectors.

$$X = |+\rangle\langle+| - |-\rangle\langle-| \qquad Y = |+i\rangle\langle+i| - |-i\rangle\langle-i| \qquad Z = |0\rangle\langle 0| - |1\rangle\langle 1|$$

For example, an X measurement is a measurement with respect to the basis $\{|+\rangle, |-\rangle\}$. Equivalently it is the measurement described by the set $\{|+\rangle\langle+|, |-\rangle\langle-|\}$.

We can perform this measurement non-destructively using *phase estimation.*



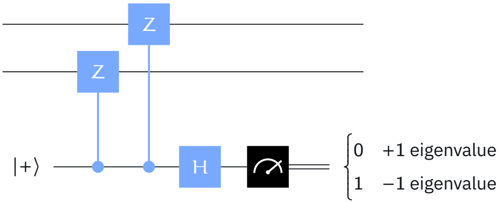
Pauli observables

This extends naturally to n-qubit Pauli operations. For example, consider $Z \otimes Z$.

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

The associated measurement is the two-outcome projective measurement described by the set $\{|00\rangle\langle 00| + |11\rangle\langle 11|, |01\rangle\langle 01| + |10\rangle\langle 10|\}$.

Again we can perform this measurement non-destructively using phase estimation.



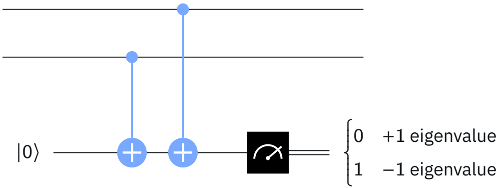
Pauli observables

This extends naturally to n-qubit Pauli operations. For example, consider $Z \otimes Z$.

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

The associated measurement is the two-outcome projective measurement described by the set $\{|00\rangle\langle 00| + |11\rangle\langle 11|, |01\rangle\langle 01| + |10\rangle\langle 10|\}$.

Again we can perform this measurement non-destructively using phase estimation.



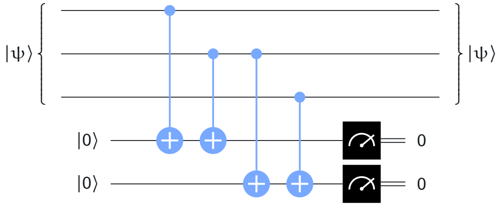
Repetition code revisited

The 3-bit repetition code encodes qubit states as follows:

$$\alpha|0\rangle + \beta|1\rangle \mapsto \alpha|000\rangle + \beta|111\rangle = |\psi\rangle$$

To check that the 3-qubit state $|\psi\rangle$ is a valid encoding of a qubit, it suffices to check these two equations:

$$(Z \otimes Z \otimes \mathbb{1})|\psi\rangle = |\psi\rangle$$
$$(\mathbb{1} \otimes Z \otimes Z)|\psi\rangle = |\psi\rangle$$



Repetition code revisited

The 3-bit repetition code encodes qubit states as follows:

$$\alpha|0\rangle + \beta|1\rangle \mapsto \alpha|000\rangle + \beta|111\rangle = |\psi\rangle$$

To check that the 3-qubit state $|\psi\rangle$ is a valid encoding of a qubit, it suffices to check these two equations:

$$(Z \otimes Z \otimes \mathbb{1})|\psi\rangle = |\psi\rangle$$
$$(\mathbb{1} \otimes Z \otimes Z)|\psi\rangle = |\psi\rangle$$

The 3-qubit Pauli operations $Z \otimes Z \otimes \mathbb{1}$ and $\mathbb{1} \otimes Z \otimes Z$ are *stabilizer generators* for this code. The *stabilizer* for the code is the set generated by the stabilizer generators.

$$\langle Z \otimes Z \otimes \mathbb{1}, \mathbb{1} \otimes Z \otimes Z\rangle = \{\mathbb{1} \otimes \mathbb{1} \otimes \mathbb{1},\ Z \otimes Z \otimes \mathbb{1},\ \mathbb{1} \otimes Z \otimes Z,\ Z \otimes \mathbb{1} \otimes Z\}$$

Bit-flip detection

$$\alpha|0\rangle + \beta|1\rangle \mapsto \alpha|000\rangle + \beta|111\rangle = |\psi\rangle$$

$$(Z \otimes Z \otimes \mathbb{1})|\psi\rangle = |\psi\rangle$$
$$(\mathbb{1} \otimes Z \otimes Z)|\psi\rangle = |\psi\rangle$$

Suppose a bit-flip error occurs on the leftmost qubit.

$$|\psi\rangle \;\mapsto\; (X \otimes \mathbb{1} \otimes \mathbb{1})|\psi\rangle$$

By treating the stabilizer generators as observables, we can detect this error.

$$(Z \otimes Z \otimes \mathbb{1})(X \otimes \mathbb{1} \otimes \mathbb{1})|\psi\rangle = -(X \otimes \mathbb{1} \otimes \mathbb{1})(Z \otimes Z \otimes \mathbb{1})|\psi\rangle = -(X \otimes \mathbb{1} \otimes \mathbb{1})|\psi\rangle$$
$$(\mathbb{1} \otimes Z \otimes Z)(X \otimes \mathbb{1} \otimes \mathbb{1})|\psi\rangle = (X \otimes \mathbb{1} \otimes \mathbb{1})(\mathbb{1} \otimes Z \otimes Z)|\psi\rangle = (X \otimes \mathbb{1} \otimes \mathbb{1})|\psi\rangle$$

$$(Z \otimes Z \otimes \mathbb{1})(X \otimes \mathbb{1} \otimes \mathbb{1}) = -(X \otimes \mathbb{1} \otimes \mathbb{1})(Z \otimes Z \otimes \mathbb{1})$$
$$(\mathbb{1} \otimes Z \otimes Z)(X \otimes \mathbb{1} \otimes \mathbb{1}) = (X \otimes \mathbb{1} \otimes \mathbb{1})(\mathbb{1} \otimes Z \otimes Z)$$

Bit-flip detection

$$\alpha|0\rangle + \beta|1\rangle \mapsto \alpha|000\rangle + \beta|111\rangle = |\psi\rangle$$

$$(Z \otimes Z \otimes \mathbb{1})|\psi\rangle = |\psi\rangle$$
$$(\mathbb{1} \otimes Z \otimes Z)|\psi\rangle = |\psi\rangle$$

Suppose a bit-flip error occurs on the leftmost qubit.

$$|\psi\rangle \;\mapsto\; (X \otimes \mathbb{1} \otimes \mathbb{1})|\psi\rangle$$

By treating the stabilizer generators as observables, we can detect this error.

$$(Z \otimes Z \otimes \mathbb{1})(X \otimes \mathbb{1} \otimes \mathbb{1}) = -(X \otimes \mathbb{1} \otimes \mathbb{1})(Z \otimes Z \otimes \mathbb{1})$$
$$(\mathbb{1} \otimes Z \otimes Z)(X \otimes \mathbb{1} \otimes \mathbb{1}) = (X \otimes \mathbb{1} \otimes \mathbb{1})(\mathbb{1} \otimes Z \otimes Z)$$

	$\mathbb{1} \otimes \mathbb{1} \otimes \mathbb{1}$	$X \otimes \mathbb{1} \otimes \mathbb{1}$	$\mathbb{1} \otimes X \otimes \mathbb{1}$	$\mathbb{1} \otimes \mathbb{1} \otimes X$
$Z \otimes Z \otimes \mathbb{1}$	$+1$	-1	-1	$+1$
$\mathbb{1} \otimes Z \otimes Z$	$+1$	$+1$	-1	-1

syndromes

Syndromes

	$\mathbb{1}\otimes\mathbb{1}\otimes\mathbb{1}$	$X\otimes\mathbb{1}\otimes\mathbb{1}$	$\mathbb{1}\otimes X\otimes\mathbb{1}$	$\mathbb{1}\otimes\mathbb{1}\otimes X$
$Z\otimes Z\otimes\mathbb{1}$	$+1$	-1	-1	$+1$
$\mathbb{1}\otimes Z\otimes Z$	$+1$	$+1$	-1	-1

syndromes

The syndromes partition the 8-dimensional space into four 2-dimensional subspaces.

$$
\begin{array}{c c c|c|c|}
 & & \multicolumn{3}{c}{\overbrace{\hspace{6em}}^{\mathbb{1}\otimes Z\otimes Z}} \\
 & & \multicolumn{1}{c}{+1} & \multicolumn{1}{c}{-1} \\
\cline{3-4}
Z\otimes Z\otimes\mathbb{1} & \left\{\begin{array}{c} +1 \\ \\ -1 \end{array}\right. & \begin{array}{c}|000\rangle \\ |111\rangle \\ \hline |100\rangle \\ |011\rangle\end{array} & \begin{array}{c}|001\rangle \\ |110\rangle \\ \hline |010\rangle \\ |101\rangle\end{array} \\
\cline{3-4}
\end{array}
$$

They also partition the 3-qubit Pauli operations into 4 equal-size collections. For example, $\mathbb{1}\otimes\mathbb{1}\otimes Z$, $Z\otimes Z\otimes Z$, and $X\otimes X\otimes X$ all cause the same syndrome $(+1, +1)$.

Syndromes

The syndromes partition the 8-dimensional space into four 2-dimensional subspaces.

$$
\begin{array}{cc|c|c|}
\multicolumn{2}{c}{} & \multicolumn{2}{c}{\overbrace{\quad +1 \qquad\qquad\quad -1 \quad}^{\mathbb{1}\otimes Z\otimes Z}} \\ \cline{3-4}
\multirow{2}{*}{$Z\otimes Z\otimes \mathbb{1}$} & +1 & \begin{array}{c}|000\rangle\\|111\rangle\end{array} & \begin{array}{c}|001\rangle\\|110\rangle\end{array} \\ \cline{3-4}
 & -1 & \begin{array}{c}|100\rangle\\|011\rangle\end{array} & \begin{array}{c}|010\rangle\\|101\rangle\end{array} \\ \cline{3-4}
\end{array}
$$

They also partition the 3-qubit Pauli operations into 4 equal-size collections. For example, $\mathbb{1}\otimes\mathbb{1}\otimes Z$, $Z\otimes Z\otimes Z$, and $X\otimes X\otimes X$ all cause the same syndrome $(+1,+1)$.

Pauli operations that commute with every stabilizer generator but are not themselves in the stabilizer act like Pauli operations on the encoded qubit.

Stabilizer codes

A set $\{P_1, \ldots, P_r\}$ of n-qubit Pauli operations are stabilizer generators for a *stabilizer code* if these properties are satisfied:

1. The stabilizer generators all *commute* with one another.

$$P_j P_k = P_k P_j \qquad \text{(for all } j, k \in \{1, \ldots, r\}\text{)}$$

2. The stabilizer generators form a *minimal generating set.*

$$P_k \notin \langle P_1, \ldots, P_{k-1}, P_{k+1}, \ldots, P_r \rangle \qquad \text{(for each } k \in \{1, \ldots, r\}\text{)}$$

3. At least one nonzero vector is fixed by all of the stabilizer generators.

$$-\mathbb{1}^{\otimes n} \notin \langle P_1, \ldots, P_r \rangle$$

The *code space* defined by the stabilizer generators contains all vectors that are fixed by all of the stabilizer generators.

$$\{|\psi\rangle : |\psi\rangle = P_1|\psi\rangle = \cdots = P_r|\psi\rangle\}$$

Examples

3-bit repetition code (bit-flips)

$$Z \otimes Z \otimes \mathbb{1}$$
$$\mathbb{1} \otimes Z \otimes Z$$

3-bit repetition code (phase-flips)

$$X \otimes X \otimes \mathbb{1}$$
$$\mathbb{1} \otimes X \otimes X$$

9-qubit Shor code

$$Z \otimes Z \otimes \mathbb{1} \otimes \mathbb{1} \otimes \mathbb{1} \otimes \mathbb{1} \otimes \mathbb{1} \otimes \mathbb{1} \otimes \mathbb{1}$$
$$\mathbb{1} \otimes Z \otimes Z \otimes \mathbb{1} \otimes \mathbb{1} \otimes \mathbb{1} \otimes \mathbb{1} \otimes \mathbb{1} \otimes \mathbb{1}$$
$$\mathbb{1} \otimes \mathbb{1} \otimes \mathbb{1} \otimes Z \otimes Z \otimes \mathbb{1} \otimes \mathbb{1} \otimes \mathbb{1} \otimes \mathbb{1}$$
$$\mathbb{1} \otimes \mathbb{1} \otimes \mathbb{1} \otimes \mathbb{1} \otimes Z \otimes Z \otimes \mathbb{1} \otimes \mathbb{1} \otimes \mathbb{1}$$
$$\mathbb{1} \otimes \mathbb{1} \otimes \mathbb{1} \otimes \mathbb{1} \otimes \mathbb{1} \otimes \mathbb{1} \otimes Z \otimes Z \otimes \mathbb{1}$$
$$\mathbb{1} \otimes \mathbb{1} \otimes \mathbb{1} \otimes \mathbb{1} \otimes \mathbb{1} \otimes \mathbb{1} \otimes \mathbb{1} \otimes Z \otimes Z$$
$$X \otimes X \otimes X \otimes X \otimes X \otimes X \otimes \mathbb{1} \otimes \mathbb{1} \otimes \mathbb{1}$$
$$\mathbb{1} \otimes \mathbb{1} \otimes \mathbb{1} \otimes X \otimes X \otimes X \otimes X \otimes X \otimes X$$

Examples

3-bit repetition code (bit-flips)

$$
\begin{array}{c}
Z\,Z\,\mathbb{1}\\
\mathbb{1}\,Z\,Z
\end{array}
$$

3-bit repetition code (phase-flips)

$$
\begin{array}{c}
X\,X\,\mathbb{1}\\
\mathbb{1}\,X\,X
\end{array}
$$

9-qubit Shor code

$$
\begin{array}{c}
Z\,Z\,\mathbb{1}\,\mathbb{1}\,\mathbb{1}\,\mathbb{1}\,\mathbb{1}\,\mathbb{1}\,\mathbb{1}\\
\mathbb{1}\,Z\,Z\,\mathbb{1}\,\mathbb{1}\,\mathbb{1}\,\mathbb{1}\,\mathbb{1}\,\mathbb{1}\\
\mathbb{1}\,\mathbb{1}\,\mathbb{1}\,Z\,Z\,\mathbb{1}\,\mathbb{1}\,\mathbb{1}\,\mathbb{1}\\
\mathbb{1}\,\mathbb{1}\,\mathbb{1}\,\mathbb{1}\,Z\,Z\,\mathbb{1}\,\mathbb{1}\,\mathbb{1}\\
\mathbb{1}\,\mathbb{1}\,\mathbb{1}\,\mathbb{1}\,\mathbb{1}\,\mathbb{1}\,Z\,Z\,\mathbb{1}\\
\mathbb{1}\,\mathbb{1}\,\mathbb{1}\,\mathbb{1}\,\mathbb{1}\,\mathbb{1}\,\mathbb{1}\,Z\,Z\\
X\,X\,X\,X\,X\,X\,\mathbb{1}\,\mathbb{1}\,\mathbb{1}\\
\mathbb{1}\,\mathbb{1}\,\mathbb{1}\,X\,X\,X\,X\,X\,X
\end{array}
$$

Examples

7-qubit Steane code

$$\begin{array}{ccccccc} Z & Z & Z & Z & \mathbb{1} & \mathbb{1} & \mathbb{1} \\ Z & Z & \mathbb{1} & \mathbb{1} & Z & Z & \mathbb{1} \\ Z & \mathbb{1} & Z & \mathbb{1} & Z & \mathbb{1} & Z \\ X & X & X & X & \mathbb{1} & \mathbb{1} & \mathbb{1} \\ X & X & \mathbb{1} & \mathbb{1} & X & X & \mathbb{1} \\ X & \mathbb{1} & X & \mathbb{1} & X & \mathbb{1} & X \end{array}$$

5-qubit code

$$\begin{array}{ccccc} X & Z & Z & X & \mathbb{1} \\ \mathbb{1} & X & Z & Z & X \\ X & \mathbb{1} & X & Z & Z \\ Z & X & \mathbb{1} & X & Z \end{array}$$

E-bit stabilizer code

$$\begin{array}{cc} Z & Z \\ X & X \end{array}$$

GHZ stabilizer code

$$\begin{array}{ccc} Z & Z & \mathbb{1} \\ \mathbb{1} & Z & Z \\ X & X & X \end{array}$$

Code space dimension

Suppose that $\{P_1, \ldots, P_r\}$ are n-qubit stabilizer generators for a stabilizer code.

1. $P_j P_k = P_k P_j$ for all $j, k \in \{1, \ldots, r\}$
2. $P_k \notin \langle P_1, \ldots, P_{k-1}, P_{k+1}, \ldots, P_r \rangle$ for each $k \in \{1, \ldots, r\}$
3. $-\mathbb{1} \notin \langle P_1, \ldots, P_r \rangle$

Theorem

The code space defined by $\{P_1, \ldots, P_r\}$ has dimension 2^{n-r}.

(Equivalently, the code defined by these generators encodes $n - r$ qubits.)

3-bit repetition code (bit-flips)

$$
\begin{matrix} Z & Z & \mathbb{1} \\ \mathbb{1} & Z & Z \end{matrix}
$$

$n = 3$ qubits
$r = 2$ stabilizer generators
$\Rightarrow 3 - 2 = 1$ encoded qubit

Code space dimension

Suppose that $\{P_1, \ldots, P_r\}$ are n-qubit stabilizer generators for a stabilizer code.

1. $P_j P_k = P_k P_j$ for all $j, k \in \{1, \ldots, r\}$
2. $P_k \notin \langle P_1, \ldots, P_{k-1}, P_{k+1}, \ldots, P_r \rangle$ for each $k \in \{1, \ldots, r\}$
3. $-\mathbb{1} \notin \langle P_1, \ldots, P_r \rangle$

Theorem

The code space defined by $\{P_1, \ldots, P_r\}$ has dimension 2^{n-r}.

(Equivalently, the code defined by these generators encodes $n - r$ qubits.)

5-qubit code

$$
\begin{array}{c}
X\,Z\,Z\,X\,\mathbb{1} \\
\mathbb{1}\,X\,Z\,Z\,X \\
X\,\mathbb{1}\,X\,Z\,Z \\
Z\,X\,\mathbb{1}\,X\,Z
\end{array}
$$

$n = 5$ qubits

$r = 4$ stabilizer generators

$\Rightarrow 5 - 4 = 1$ encoded qubit

Code space dimension

Suppose that $\{P_1, \ldots, P_r\}$ are n-qubit stabilizer generators for a stabilizer code.

1. $P_j P_k = P_k P_j$ for all $j, k \in \{1, \ldots, r\}$
2. $P_k \notin \langle P_1, \ldots, P_{k-1}, P_{k+1}, \ldots, P_r \rangle$ for each $k \in \{1, \ldots, r\}$
3. $-\mathbb{1} \notin \langle P_1, \ldots, P_r \rangle$

Theorem

The code space defined by $\{P_1, \ldots, P_r\}$ has dimension 2^{n-r}.

(Equivalently, the code defined by these generators encodes $n - r$ qubits.)

E-bit stabilizer code

$$\begin{array}{c} Z\,Z \\ X\,X \end{array}$$

$n = 2$ qubits

$r = 2$ stabilizer generators

$\Rightarrow 2 - 2 = 0$ encoded qubits

Clifford operations and encodings

Clifford operations

Clifford operations are unitary operations that can be implemented by quantum circuits with gates from this list:

- Hadamard gates
- S gates
- CNOT gates

Up to a global phase, an n-qubit unitary operation is a Clifford operation if and only if it maps n-qubit Pauli operations to n-qubit Pauli operations by conjugation.

Equivalently, U is a Clifford operation (up to a global phase) if for every $P_0, \ldots, P_{n-1} \in \{\mathbb{1}, X, Y, Z\}$ there exist $Q_0, \ldots, Q_{n-1} \in \{\mathbb{1}, X, Y, Z\}$ such that

$$U(P_{n-1} \otimes \cdots \otimes P_0)U^\dagger = \pm Q_{n-1} \otimes \cdots \otimes Q_0$$

Clifford operations are *not universal* for quantum computation.

There are only finitely many n-qubit Clifford operations and their actions on standard basis states can be efficiently simulated classically by the *Gottesman–Knill theorem.*

Clifford operations and encodings

Clifford operations

Clifford operations are unitary operations that can be implemented by quantum circuits with gates from this list:

- Hadamard gates
- S gates
- CNOT gates

Clifford operations are *not universal* for quantum computation.

There are only finitely many n-qubit Clifford operations and their actions on standard basis states can be efficiently simulated classically by the *Gottesman–Knill theorem.*

Encodings for stabilizer codes can always be performed using $O(n^2/\log(n))$ Clifford gates.

Clifford operations and encodings

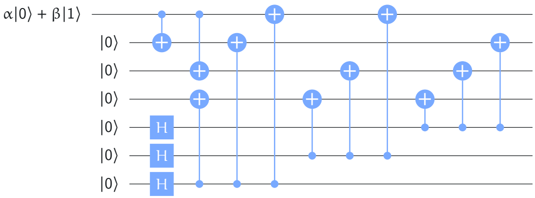
Clifford operations

Clifford operations are unitary operations that can be implemented by quantum circuits with gates from this list:

- Hadamard gates
- S gates
- CNOT gates

Encodings for stabilizer codes can always be performed using $O(n^2/\log(n))$ Clifford gates.

Example: encoder for the 7-qubit Steane code



Detecting errors

Let $P_1, \ldots, P_r$ be stabilizer generators for an n-qubit stabilizer code, and let E be an n-qubit Pauli operation, representing a *hypothetical error.*

Errors are detected in a stabilizer code by *measuring the stabilizer generators* (as observables). The r outcomes form the syndrome.

Case 1: $E = \alpha Q$ for $Q \in \langle P_1, \ldots, P_r \rangle$.

This error *does nothing* to vectors in the code space: $E|\psi\rangle = \alpha|\psi\rangle$ for every encoded state $|\psi\rangle$.

Case 2: $E \neq \alpha Q$ for $Q \in \langle P_1, \ldots, P_r \rangle$, but $E P_k = P_k E$ for every $k \in \{1, \ldots, r\}$.

This error changes vectors in the code space and goes *undetected* by the code.

Case 3: $P_k E = -E P_k$ for at least one $k \in \{1, \ldots, r\}$.

This error is *detected* by the code.

The *distance* of a stabilizer code is the *minimum weight* of a Pauli operation that changes vectors in the code space but goes undetected by the code.

Notation: an $[[n, m, d]]$ stabilizer code is one that encodes m qubits into n qubits and has distance d.

7-qubit Steane code

$$
\begin{array}{ccccccc}
Z & Z & Z & Z & \mathbb{1} & \mathbb{1} & \mathbb{1} \\
Z & Z & \mathbb{1} & \mathbb{1} & Z & Z & \mathbb{1} \\
Z & \mathbb{1} & Z & \mathbb{1} & Z & \mathbb{1} & Z \\
X & X & X & X & \mathbb{1} & \mathbb{1} & \mathbb{1} \\
X & X & \mathbb{1} & \mathbb{1} & X & X & \mathbb{1} \\
X & \mathbb{1} & X & \mathbb{1} & X & \mathbb{1} & X
\end{array}
$$

The *distance* is the minimum weight of an n-qubit Pauli operation that

1. commutes with every stabilizer generator, and
2. is not proportional to a stabilizer element.

This code has distance 3.

We can first reason that every Pauli operation with weight at most 2 that commutes with every stabilizer generator must be the identity operation.

$$
P \;\; Q \;\; \mathbb{1} \;\; \mathbb{1} \;\; \mathbb{1} \;\; \mathbb{1} \;\; \mathbb{1}
$$

$$
\begin{array}{ccccccc}
Z & \mathbb{1} & Z & \mathbb{1} & Z & \mathbb{1} & Z \\
X & \mathbb{1} & X & \mathbb{1} & X & \mathbb{1} & X
\end{array}
$$

7-qubit Steane code

$$
\begin{matrix}
Z & Z & Z & Z & \mathbb{1} & \mathbb{1} & \mathbb{1}\\
Z & Z & \mathbb{1} & \mathbb{1} & Z & Z & \mathbb{1}\\
Z & \mathbb{1} & Z & \mathbb{1} & Z & \mathbb{1} & Z\\
X & X & X & X & \mathbb{1} & \mathbb{1} & \mathbb{1}\\
X & X & \mathbb{1} & \mathbb{1} & X & X & \mathbb{1}\\
X & \mathbb{1} & X & \mathbb{1} & X & \mathbb{1} & X
\end{matrix}
$$

The *distance* is the minimum weight of an n-qubit Pauli operation that

1. commutes with every stabilizer generator, and
2. is not proportional to a stabilizer element.

This code has distance 3.

We can first reason that every Pauli operation with weight at most 2 that commutes with every stabilizer generator must be the identity operation.

$$
\mathbb{1} \quad Q \quad \mathbb{1} \quad \mathbb{1} \quad \mathbb{1} \quad \mathbb{1} \quad \mathbb{1}
$$

$$
\begin{matrix}
Z & \mathbb{1} & Z & \mathbb{1} & Z & \mathbb{1} & Z\\
X & \mathbb{1} & X & \mathbb{1} & X & \mathbb{1} & X
\end{matrix}
$$

7-qubit Steane code

$$\begin{array}{ccccccc} Z & Z & Z & Z & \mathbb{1} & \mathbb{1} & \mathbb{1} \\ Z & Z & \mathbb{1} & \mathbb{1} & Z & Z & \mathbb{1} \\ Z & \mathbb{1} & Z & \mathbb{1} & Z & \mathbb{1} & Z \\ X & X & X & X & \mathbb{1} & \mathbb{1} & \mathbb{1} \\ X & X & \mathbb{1} & \mathbb{1} & X & X & \mathbb{1} \\ X & \mathbb{1} & X & \mathbb{1} & X & \mathbb{1} & X \end{array}$$

The *distance* is the minimum weight of an n-qubit Pauli operation that

1. commutes with every stabilizer generator, and
2. is not proportional to a stabilizer element.

This code has distance 3.

We can first reason that every Pauli operation with weight at most 2 that commutes with every stabilizer generator must be the identity operation.

$$\begin{array}{ccccccc} \mathbb{1} & Q & \mathbb{1} & \mathbb{1} & \mathbb{1} & \mathbb{1} & \mathbb{1} \end{array}$$

$$\begin{array}{ccccccc} Z & \mathbb{1} & Z & \mathbb{1} & Z & \mathbb{1} & Z \\ X & \mathbb{1} & X & \mathbb{1} & X & \mathbb{1} & X \\ Z & Z & Z & Z & \mathbb{1} & \mathbb{1} & \mathbb{1} \\ X & X & X & X & \mathbb{1} & \mathbb{1} & \mathbb{1} \end{array}$$

7-qubit Steane code

$$
\begin{matrix}
Z & Z & Z & Z & \mathbb{1} & \mathbb{1} & \mathbb{1} \\
Z & Z & \mathbb{1} & \mathbb{1} & Z & Z & \mathbb{1} \\
Z & \mathbb{1} & Z & \mathbb{1} & Z & \mathbb{1} & Z \\
X & X & X & X & \mathbb{1} & \mathbb{1} & \mathbb{1} \\
X & X & \mathbb{1} & \mathbb{1} & X & X & \mathbb{1} \\
X & \mathbb{1} & X & \mathbb{1} & X & \mathbb{1} & X
\end{matrix}
$$

The *distance* is the minimum weight of an n-qubit Pauli operation that

1. commutes with every stabilizer generator, and
2. is not proportional to a stabilizer element.

This code has distance 3.

We can first reason that every Pauli operation with weight at most 2 that commutes with every stabilizer generator must be the identity operation.

$$
\begin{matrix}
\mathbb{1} & \mathbb{1} & \mathbb{1} & \mathbb{1} & \mathbb{1} & \mathbb{1} & \mathbb{1}
\end{matrix}
$$

$$
\begin{matrix}
Z & \mathbb{1} & Z & \mathbb{1} & Z & \mathbb{1} & Z \\
X & \mathbb{1} & X & \mathbb{1} & X & \mathbb{1} & X \\
Z & Z & Z & Z & \mathbb{1} & \mathbb{1} & \mathbb{1} \\
X & X & X & X & \mathbb{1} & \mathbb{1} & \mathbb{1}
\end{matrix}
$$

7-qubit Steane code

$$
\begin{array}{ccccccc}
Z & Z & Z & Z & \mathbb{1} & \mathbb{1} & \mathbb{1} \\
Z & Z & \mathbb{1} & \mathbb{1} & Z & Z & \mathbb{1} \\
Z & \mathbb{1} & Z & \mathbb{1} & Z & \mathbb{1} & Z \\
X & X & X & X & \mathbb{1} & \mathbb{1} & \mathbb{1} \\
X & X & \mathbb{1} & \mathbb{1} & X & X & \mathbb{1} \\
X & \mathbb{1} & X & \mathbb{1} & X & \mathbb{1} & X
\end{array}
$$

The *distance* is the minimum weight of an n-qubit Pauli operation that

1. commutes with every stabilizer generator, and
2. is not proportional to a stabilizer element.

This code has distance 3. ✓

We can first reason that every Pauli operation with weight at most 2 that commutes with every stabilizer generator must be the identity operation.

On the other hand, there are weight 3 Pauli operations that commute with every stabilizer generator and fall outside of the stabilizer.

Two examples:

$$
\begin{array}{ccccccc}
\mathbb{1} & \mathbb{1} & \mathbb{1} & \mathbb{1} & X & X & X \\
\mathbb{1} & \mathbb{1} & \mathbb{1} & \mathbb{1} & Z & Z & Z
\end{array}
$$

Correcting errors

Let $P_1, \ldots, P_r$ be stabilizer generators for an n-qubit stabilizer code.

- The 2^r syndromes partition the n-qubit Pauli operations into equal-size sets, with $4^n/2^r$ Pauli operations in each set.
- If E is an error and $S \in \langle P_1, \ldots, P_r \rangle$ is a stabilizer element, then E and ES are equivalent errors: $E|\psi\rangle = ES|\psi\rangle$ for every $|\psi\rangle$ in the code space.
- This leaves 4^{n-r} inequivalent classes of errors for each syndrome.

So, unless $r = n$ (i.e., the code space is one-dimensional) we cannot correct every error. Rather, we must choose *one correction operation for each syndrome* (which corrects at most one class of equivalent errors).

Natural strategy

For each syndrome s, choose a *lowest weight* Pauli operation that causes the syndrome s as the corresponding correction operation.

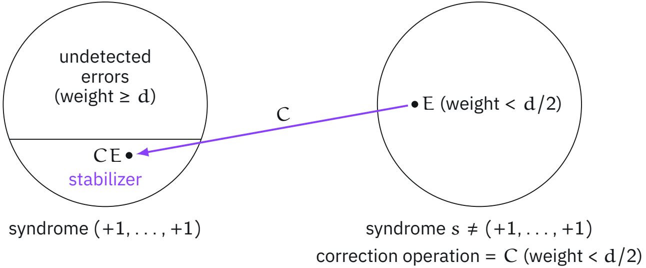
For a distance d stabilizer code, this strategy corrects all errors having weight at most $(d-1)/2$.

Correcting errors

Natural strategy

For each syndrome s, choose a *lowest weight* Pauli operation that causes the syndrome s as the corresponding correction operation.

For a distance d stabilizer code, this strategy corrects all errors having weight at most $(d - 1)/2$.



Correcting errors

Natural strategy

For each syndrome s, choose a *lowest weight* Pauli operation that causes the syndrome s as the corresponding correction operation.

For a distance d stabilizer code, this strategy corrects all errors having weight at most $(\mathrm{d}-1)/2$.

Unfortunately, for a given choice of stabilizer generators and a syndrome, it is *computationally difficult* to find the lowest weight Pauli operation causing that syndrome.

Finding codes for which this can be done efficiently is part of the artistry in code design.

Classical linear codes

Let $\Sigma = \{0, 1\}$ denote the binary alphabet.

A *classical linear code* is a non-empty set of binary strings $\mathcal{C} \subseteq \Sigma^n$ with this property:

$$u, v \in \mathcal{C} \implies u \oplus v \in \mathcal{C}$$

Example: 3-bit repetition code

The 3-bit repetition code $\{000, 111\}$ is a classical linear code.

Example: $[7, 4, 3]$-Hamming code

The $[7, 4, 3]$-Hamming code is the classical linear code containing these strings:

$$\begin{array}{cccc}
0000000 & 1100001 & 1010010 & 0110011 \\
0110100 & 1010101 & 1100110 & 0000111 \\
1111000 & 0011001 & 0101010 & 1001011 \\
1001100 & 0101101 & 0011110 & 1111111
\end{array}$$

Classical linear codes

Let $\Sigma = \{0, 1\}$ denote the binary alphabet.

A *classical linear code* is a non-empty set of binary strings $\mathcal{C} \subseteq \Sigma^n$ with this property:

$$u, v \in \mathcal{C} \;\Rightarrow\; u \oplus v \in \mathcal{C}$$

Two natural ways to describe a classical linear code:

1. *Generators:* a minimal list of strings $u_1, \ldots, u_m \in \Sigma^n$ such that

$$\mathcal{C} = \{\alpha_1 u_1 \oplus \cdots \oplus \alpha_m u_m \,:\, \alpha_1, \ldots, \alpha_m \in \{0, 1\}\}$$

2. *Parity checks:* a minimal list of strings $v_1, \ldots, v_r \in \Sigma^n$ such that

$$\mathcal{C} = \{u \in \Sigma^n \,:\, u \cdot v_1 = \cdots = u \cdot v_r = 0\}$$

(where $u \cdot v$ is the binary dot product of u and v).

Classical linear codes

1. *Generators:* a minimal list of strings $u_1, \ldots, u_m \in \Sigma^n$ such that

$$\mathcal{C} = \{\alpha_1 u_1 \oplus \cdots \oplus \alpha_m u_m \,:\, \alpha_1, \ldots, \alpha_m \in \{0,1\}\}$$

2. *Parity checks:* a minimal list of strings $v_1, \ldots, v_r \in \Sigma^n$ such that

$$\mathcal{C} = \{u \in \Sigma^n \,:\, u \cdot v_1 = \cdots = u \cdot v_r = 0\}$$

Example: 3-bit repetition code

The 3-bit repetition code $\{000, 111\}$ is a classical linear code.

1. Generator: 111
2. Parity checks: 110, 011

Classical linear codes

1. *Generators:* a minimal list of strings $u_1, \ldots, u_m \in \Sigma^n$ such that

$$\mathcal{C} = \{\alpha_1 u_1 \oplus \cdots \oplus \alpha_m u_m \, : \, \alpha_1, \ldots, \alpha_m \in \{0,1\}\}$$

2. *Parity checks:* a minimal list of strings $v_1, \ldots, v_r \in \Sigma^n$ such that

$$\mathcal{C} = \{u \in \Sigma^n \, : \, u \cdot v_1 = \cdots = u \cdot v_r = 0\}$$

Example: $[7,4,3]$-Hamming code

The $[7,4,3]$-Hamming code is the classical linear code containing these strings:

$$\begin{array}{cccc} 0000000 & 1100001 & 1010010 & 0110011 \\ 0110100 & 1010101 & 1100110 & 0000111 \\ 1111000 & 0011001 & 0101010 & 1001011 \\ 1001100 & 0101101 & 0011110 & 1111111 \end{array}$$

1. Generators: 0110100, 1010010, 1100001, 1111000
2. Parity checks: 1111000, 1100110, 1010101

Classical linear codes

1. *Generators:* a minimal list of strings $u_1, \ldots, u_m \in \Sigma^n$ such that

$$\mathcal{C} = \{\alpha_1 u_1 \oplus \cdots \oplus \alpha_m u_m : \alpha_1, \ldots, \alpha_m \in \{0,1\}\}$$

2. *Parity checks:* a minimal list of strings $v_1, \ldots, v_r \in \Sigma^n$ such that

$$\mathcal{C} = \{u \in \Sigma^n : u \cdot v_1 = \cdots = u \cdot v_r = 0\}$$

Note: parity checks are equivalent to *stabilizer generators* containing only Z and $\mathbb{1}$ Pauli matrices.

Example: 3-bit repetition code

The 3-bit repetition code $\{000, 111\}$ is a classical linear code.

1. Generator: 111
2. Parity checks: 110, 011

Equivalently, the strings in this code are standard basis states for the stabilizer code with stabilizer generators $Z\,Z\,\mathbb{1}$ and $\mathbb{1}\,Z\,Z$.

CSS codes

Stabilizer generators containing only Z and $\mathbb{1}$ Pauli matrices are equivalent to parity checks.

Example: 3-bit repetition code

The 3-bit repetition code $\{000, 111\}$ is a classical linear code.

Parity checks: 110, 011
Stabilizer generators: $Z\,Z\,\mathbb{1}$, $\mathbb{1}\,Z\,Z$

Example: $[7,4,3]$-Hamming code

The $[7,4,3]$-Hamming code is the classical linear code containing these strings:

$$
\begin{array}{cccc}
0000000 & 1100001 & 1010010 & 0110011 \\
0110100 & 1010101 & 1100110 & 0000111 \\
1111000 & 0011001 & 0101010 & 1001011 \\
1001100 & 0101101 & 0011110 & 1111111
\end{array}
$$

Parity checks: 1111000, 1100110, 1010101
Stabilizer generators: $Z\,Z\,Z\,Z\,\mathbb{1}\,\mathbb{1}\,\mathbb{1}$, $Z\,Z\,\mathbb{1}\,\mathbb{1}\,Z\,Z\,\mathbb{1}$, $Z\,\mathbb{1}\,Z\,\mathbb{1}\,Z\,\mathbb{1}\,Z$

CSS codes

Stabilizer generators containing only Z and $\mathbb{1}$ Pauli matrices are equivalent to parity checks. These are called *Z stabilizer generators.*

Stabilizer generators containing only X and $\mathbb{1}$ Pauli matrices are also equivalent to parity checks — for the plus/minus basis $\{|+\rangle, |-\rangle\}$.

Example: $[7,4,3]$-Hamming code

$$\begin{array}{cccc} 0000000 & 1100001 & 1010010 & 0110011 \\ 0110100 & 1010101 & 1100110 & 0000111 \\ 1111000 & 0011001 & 0101010 & 1001011 \\ 1001100 & 0101101 & 0011110 & 1111111 \end{array}$$

Parity checks: 1111000, 1100110, 1010101

The stabilizer generators $X\,X\,X\,X\,\mathbb{1}\,\mathbb{1}\,\mathbb{1}$, $X\,X\,\mathbb{1}\,\mathbb{1}\,X\,X\,\mathbb{1}$, $X\,\mathbb{1}\,X\,\mathbb{1}\,X\,\mathbb{1}\,X$ define a code that includes these states:

$$\begin{array}{cccc} |{+}{+}{+}{+}{+}{+}{+}\rangle & |{-}{-}{+}{+}{+}{+}{-}\rangle & |{-}{+}{-}{+}{+}{-}{+}\rangle & |{+}{-}{-}{+}{+}{-}{-}\rangle \\ |{+}{-}{-}{+}{-}{+}{+}\rangle & |{-}{+}{-}{+}{-}{+}{-}\rangle & |{-}{-}{+}{+}{-}{-}{+}\rangle & |{+}{+}{+}{+}{-}{-}{-}\rangle \\ |{-}{-}{-}{-}{+}{+}{+}\rangle & |{+}{+}{-}{-}{+}{+}{-}\rangle & |{+}{-}{+}{-}{+}{-}{+}\rangle & |{-}{+}{+}{-}{+}{-}{-}\rangle \\ |{-}{+}{+}{-}{-}{+}{+}\rangle & |{+}{-}{+}{-}{-}{+}{-}\rangle & |{+}{+}{-}{-}{-}{-}{+}\rangle & |{-}{-}{-}{-}{-}{-}{-}\rangle \end{array}$$

CSS codes

Stabilizer generators containing only Z and $\mathbb{1}$ Pauli matrices are equivalent to parity checks. These are called *Z stabilizer generators.*

Stabilizer generators containing only X and $\mathbb{1}$ Pauli matrices are also equivalent to parity checks — for the plus/minus basis $\{|+\rangle, |-\rangle\}$. These are called *X stabilizer generators.*

Definition: CSS codes

Stabilizer codes that can be expressed using only Z stabilizer generators and X stabilizer generators are called *CSS codes.*

Example: e-bit stabilizer code

$$\begin{matrix} Z\,Z \\ X\,X \end{matrix}$$

The code space is the one-dimensional space spanned by

$$|\phi^+\rangle = \frac{|0\rangle|0\rangle + |1\rangle|1\rangle}{\sqrt{2}} = \frac{|+\rangle|+\rangle + |-\rangle|-\rangle}{\sqrt{2}}$$

CSS codes

Stabilizer generators containing only Z and $\mathbb{1}$ Pauli matrices are equivalent to parity checks. These are called *Z stabilizer generators.*

Stabilizer generators containing only X and $\mathbb{1}$ Pauli matrices are also equivalent to parity checks — for the plus/minus basis $\{|+\rangle, |-\rangle\}$. These are called *X stabilizer generators.*

Definition: CSS codes

Stabilizer codes that can be expressed using only Z stabilizer generators and X stabilizer generators are called *CSS codes.*

Example: 7-qubit Steane code

$$
\begin{array}{ccccccc}
Z & Z & Z & Z & \mathbb{1} & \mathbb{1} & \mathbb{1}\\
Z & Z & \mathbb{1} & \mathbb{1} & Z & Z & \mathbb{1}\\
Z & \mathbb{1} & Z & \mathbb{1} & Z & \mathbb{1} & Z\\
X & X & X & X & \mathbb{1} & \mathbb{1} & \mathbb{1}\\
X & X & \mathbb{1} & \mathbb{1} & X & X & \mathbb{1}\\
X & \mathbb{1} & X & \mathbb{1} & X & \mathbb{1} & X
\end{array}
$$

Example: 9-qubit Shor code

$$
\begin{array}{ccccccccc}
Z & Z & \mathbb{1} & \mathbb{1} & \mathbb{1} & \mathbb{1} & \mathbb{1} & \mathbb{1} & \mathbb{1}\\
\mathbb{1} & Z & Z & \mathbb{1} & \mathbb{1} & \mathbb{1} & \mathbb{1} & \mathbb{1} & \mathbb{1}\\
\mathbb{1} & \mathbb{1} & \mathbb{1} & Z & Z & \mathbb{1} & \mathbb{1} & \mathbb{1} & \mathbb{1}\\
\mathbb{1} & \mathbb{1} & \mathbb{1} & \mathbb{1} & Z & Z & \mathbb{1} & \mathbb{1} & \mathbb{1}\\
\mathbb{1} & \mathbb{1} & \mathbb{1} & \mathbb{1} & \mathbb{1} & \mathbb{1} & Z & Z & \mathbb{1}\\
\mathbb{1} & \mathbb{1} & \mathbb{1} & \mathbb{1} & \mathbb{1} & \mathbb{1} & \mathbb{1} & Z & Z\\
X & X & X & X & X & X & \mathbb{1} & \mathbb{1} & \mathbb{1}\\
\mathbb{1} & \mathbb{1} & \mathbb{1} & X & X & X & X & X & X
\end{array}
$$

Error detection and correction

Consider a CSS code.

- The Z stabilizer generators detect X errors but are oblivious to Z errors (and corrections).
- The X stabilizer generators detect Z errors but are oblivious to X errors (and corrections).

Suppose the following:

- The Z stabilizer generators allow for the correction of up to j bit-flip errors.
- The X stabilizer generators allow for the correction of up to k phase-flip errors.

7-qubit Steane code

$$
\begin{array}{ccccccc}
Z & Z & Z & Z & \mathbb{1} & \mathbb{1} & \mathbb{1} \\
Z & Z & \mathbb{1} & \mathbb{1} & Z & Z & \mathbb{1} \\
Z & \mathbb{1} & Z & \mathbb{1} & Z & \mathbb{1} & Z \\
X & X & X & X & \mathbb{1} & \mathbb{1} & \mathbb{1} \\
X & X & \mathbb{1} & \mathbb{1} & X & X & \mathbb{1} \\
X & \mathbb{1} & X & \mathbb{1} & X & \mathbb{1} & X
\end{array}
$$

Then the CSS code allows for the correction of *any error* on up to $\min\{j, k\}$ qubits — we can simply detect and correct X errors and Z errors on this many qubits separately.

Thank you for your attention!