Chain Complexes + CSS codes (§7.18)

CSS codes are closely related to chain complexes. This relationship can be helpful for discovering codes, understanding their properties, thinking about decoding, and doing calculations. The chain complex viewpoint also illuminates the structure of hypergraph product codes.

We'll understand the connection by revisiting the toric code, and then generalizing.

A chain complex is a sequence of vector spaces and "boundary operators" mapping each space to the next one in the sequence

$$\cdots \longrightarrow V_2 \xrightarrow{\partial_2} V_1 \xrightarrow{\partial_1} V_0 \longrightarrow$$

where e.g. $\partial_1 \circ \partial_2 = 0$ (the boundary of a boundary is zero.) In the case of interest to us, the vector spaces are over $\mathbb{F}_2$. V_2 is associated with the elementary plaquettes on a lattice (a.k.a faces a.k.a. 2-cells): Each 2-cell is assigned either 0 or 1. V_1 is likewise associated with edges (1-cells) and V_0 with vertices (0-cells).

We'll identify elements of V_2 (2-chains) with elements of the Z-type stabilizer of the toric code. Each check operator (S_Z generator) is Z_p, product of $Z^{\otimes 4}$ on plaquette p. An element of S_Z is

$$\tilde{Z}(w) = \prod_p Z_p^{w_p}$$

w_p in the 2-chain, assigns 0 or 1 to each plaquette.

22 Apr 2026

Then multiplication of stabilizers becomes addition of vectors over $\mathbb{F}_2$

$$\tilde{Z}(\omega)\,\tilde{Z}(\omega') = \tilde{Z}(\omega+\omega')$$

An arbitrary Z-type operator corresponds to a 1-chain, which assigns 0 or 1 to each lattice link.

$$Z(\varepsilon) = \bigotimes_e Z_e^{\varepsilon_e}$$

Tensor product over all lattice links – i.e. all qubits in code block.

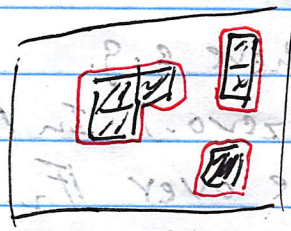
$$Z(\varepsilon)\,Z(\varepsilon') = Z(\varepsilon+\varepsilon')$$

Boundary operator $\partial_2 : V_2 \to V_1$

$$(\partial_2 \omega)_\ell = \text{sum mod 2} \sum_{p \supset \ell} \omega_p$$

$$\Rightarrow \quad \tilde{Z}(\omega) = Z(\partial_2 \omega)$$

Sum of boundaries of droplets of filled plaquettes



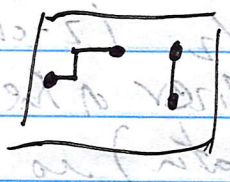
Links inside droplet cancel

Another boundary operator

$\partial_1 : V_1 \to V_0$ (edges → vertices)

Assigns to site s

$$(\partial \varepsilon)_s = \sum_{\ell \ni s} \varepsilon_\ell$$



Sum of endpoints of connected paths

Boundaries are defined so $\partial_1 \circ \partial_2 = 0$

(Boundaries of droplets are closed loops.)

22 Apr 2026

When does a Z-type operator commute with S_X (so contained in $S_X^\perp$)?

At each site, $Z(\mathcal{E})$ contains even number of Z's meeting at that site

Hence $\partial_1 \mathcal{E} = 0$ "$\mathcal{E}$ is a 1-cycle"

$\partial_1 \circ \partial_2 = 0$ means elements of the Z-type stabilizer commute with X-type stabilizer (as required for CSS code).

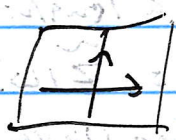
Z-type logical operators: $Z(\mathcal{E})$ where $\mathcal{E}$ is a 1-cycle, trivial Z-type logical operators (in stabilizer) are those in image of ∂_2 (in Z-type stabilizer).

$$V_2 \xrightarrow{\partial_2} V_1 \xrightarrow{\partial_1} V_0$$

Z-stab: $\tilde{Z}(\omega)$ $\quad$ $Z(\ell)$ $\quad$ X-type syndrome $X(s)$

$$\mathcal{L}_Z = \frac{\ker(\partial_1)}{\mathrm{Im}(\partial_2)} \equiv H_1(C) \qquad C = \text{chain complex over } \mathbb{F}_2$$

For Torus $\quad H_1(C) = \mathbb{Z}_2 \oplus \mathbb{Z}_2$



↑ horizontal cycle $\quad$ ↖ vertical cycle

What about X-type logical operators? We could use the same construction on the dual lattice. Or — consider the dual complex in which coboundary operators act in the opposite direction. These are transposeds of boundary operators

22 Apr 2026

$$V_2 \underset{\delta_1}{\overset{\partial_2}{\rightleftarrows}} V_1 \underset{\delta_0}{\overset{\partial_1}{\rightleftarrows}} V_0 \qquad \delta_1 = \partial_2^T \qquad \delta_0 = \partial_1^T$$

δ_0 maps site to all links meeting at site.

δ_1 maps link to all plaquettes meeting at link.

$$\delta_1 \circ \delta_0 = \partial_2^T \circ \partial_1^T = (\partial_1 \circ \partial_2)^T = 0$$

$$V_2 \xleftarrow{\delta_1} V_1 \xleftarrow{\delta_0} V_0$$

$Z_{stab} \leftarrow$ qubits $\rightarrow X_{stab}$

X-Type logicals are in Kernel of δ_1 (commute with Z-type stabilizers); ~~trivial~~ trivial if in the image of δ_0 (contained in stabilizer).

$$\mathcal{L}_X = \frac{\ker(\delta_1)}{\mathrm{Im}(\delta_0)} = H^1(C^*) = \mathbb{Z}_2 \oplus \mathbb{Z}_2$$

General CSS code

Z-type H_Z is ~~$m_Z \times n$~~ $m_Z \times n$

X-type H_X is $m_X \times n$

General Z-type operator is $Z(\xi)$, ξ a length-n 1-chain

Z-Type stabilizer = product of generators = $\tilde{Z}(\omega)$

ω is a length-m_Z 2-chain, indicating which generators (rows of H_Z) are multiplied together.

The boundary $\partial_2 \omega$ of ω indicates which qubits support Z when the generators are multiplied. We sum the rows of H_Z that are tagged by ω (columns of H_Z^T)

$$\partial_2 \omega = H_Z^T \omega \quad \text{(length } n\text{)}$$

22 Apr 2026

For the 1-chain $\mathcal{E}$, we define its boundary as its length-m_X X-type stabilizer

$$\partial_1 \mathcal{E} = H_X \mathcal{E}$$

Then $\partial_1 \circ \partial_2 = H_X H_Z^T = 0$ for a CSS code.

Z stabilizers have trivial X-type syndrome

$$\mathcal{L}_Z = \frac{\ker(\partial_1)}{\mathrm{Im}(\partial_2)} = \frac{\ker(H_X)}{\mathrm{Im}(H_Z^T)}$$

For the dual complex

$$\delta_1 = \partial_2^T = H_Z \qquad \delta_0 = \partial_1^T = H_X^T$$

$$\delta_1 \circ \delta_0 = H_Z H_X^T = 0$$

$$\mathcal{L}_X = \frac{\ker(\delta_1)}{\mathrm{Im}(\delta_0)} = \frac{\ker(H_Z)}{\mathrm{Im}(H_X^T)}$$

Check: Dimensions match for $\mathcal{L}_Z$ and $\mathcal{L}_X$

$$\dim \ker H_X - \dim \mathrm{Im}\, H_Z^T$$
$$= \dim \ker H_Z - \dim \mathrm{Im}\, H_X^T$$

Recall: $\dim \ker H + \dim \mathrm{Im}\, H = n$ if H is $r \times n$

This is rank-nullity thm: Every dimension either annihilated (kernel) or mapped to output (image).

Also: $\dim \mathrm{Im}\, H = \dim \mathrm{Im}\, H^T$ (number of linearly independent rows = number of linearly independent columns (e.g. show by Gaussian elimination).

Hence $\dim \ker H_X + \dim \mathrm{Im}\, H_X^T = n = \dim \ker H_Z + \dim \mathrm{Im}\, H_Z^T$ - it checks.

Homology / Cohomology also provide a pairing of elements of $\mathcal{L}_Z$ and $\mathcal{L}_X$ (Poincare duality), the intersection product, which define a logical basis. Cycles and Cocycles "pair" one another. We saw how this works for the toric code, and it holds more generally.

We can understand the hypergraph product as a tensor product of chain complexes. First: a classical code and its transpose in language of homology.

$$0 \xrightarrow{\partial_2} V_1 \xrightarrow{\partial_1} V_0 \xrightarrow{\partial_0} 0$$

∂_2 has trivial kernel, Kernel of ∂_0 is all of V_0

$\partial_1 = H$ is parity check matrix

$H_1(C) = \mathcal{L} = \dfrac{\ker \partial_1}{\mathrm{Im}\, \partial_2} = \ker H$. that is the code space

Also of interest:

$H_0(C) = \mathcal{L}^T = \dfrac{\ker \partial_0}{\mathrm{Im}\, \partial_1} = \dfrac{V_0}{\mathrm{Im}\, H}$

these are syndromes that don't occur because H is redundant (more rows than needed)

The vectors correspond to linear combinations of rows of H that sum to zero. $v^T H = 0 \Rightarrow H^T v = 0$

these are codewords of transpose code.

$H_1(C)$ is code, $H_0(C)$ is transpose code.

In HGP we put two chain complexes together to get a product complex that defines a CSS code.

$$C_A: \quad 0 \xrightarrow{(\partial_2)_A} A_1 \xrightarrow{(\partial_1)_A} A_0 \xrightarrow{(\partial_0)_A} 0 \qquad (\partial_1)_A = A \text{ parity check.}$$

$$C_B: \quad 0 \xrightarrow{(\partial_2)_B} B_1 \xrightarrow{(\partial_1)_B} B_0 \xrightarrow{(\partial_0)_B} 0$$

In the product complex

$$C_A \otimes C_B: \quad A_1 \otimes B_1 \xrightarrow{\partial_2} A_1 \otimes B_0 \oplus A_0 \otimes B_1 \xrightarrow{\partial_1} A_0 \otimes B_0$$

[qubits in two registers.

Boundary acts by product rule:

$$\partial_2 (A_1 \otimes B_1) = (\partial_1)_A A_1 \otimes B_1 + A_1 \otimes (\partial_1)_B B_1$$

$$\partial_1 (A_1 \otimes B_0 \oplus A_0 \otimes B_1)$$

$$= \partial_1 A_1 \otimes B_0 + A_1 \otimes (\partial_0)_B B_0 + \partial_0 A_0 \otimes B_1 + A_0 \otimes \partial_1 B_1$$

check $\partial_1 \circ \partial_2 = 0$

$$\partial_1 (\partial_2 (A_1 \otimes B_1)) = \partial_0 \partial_1 A_1 \otimes B_1 + \partial_1 A_1 \otimes \partial_1 B_1 + \partial_1 A_1 \otimes \partial_1 B_1 + A_1 \otimes \partial_0 \partial_1 B_1$$

$\partial_0 \partial_1 = 0 \Rightarrow$ $= \partial_1 A_1 \otimes \partial_1 B_1 + \partial_1 A_1 \otimes \partial_1 B_1 = 0$ (over $\mathbb{F}_2$).

This is the homological way to understand our observation that HGP arranges for collisions between H_X and H_Z to occur in matched pairs

With this product complex, we construct X-type (or Z-type) logical operators of a CSS code.

This allows to draw on the rich mathematical knowledge about homological algebra. Example: Topologists know the Künneth formula for homology of product complexes, in particular:

$$H_1(A \otimes B) = H_1(A) \otimes H_0(B) \oplus H_0(A) \otimes H_1(B)$$

(A code, B^T code, A^T code, B code)

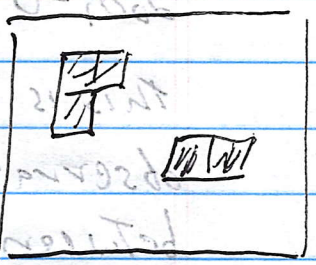
In particular, $k = k_A k_{B^T} + k_{A^T} k_B$

[To match this to what I claimed earlier, need to flip $B \to B^T$. Sorry about that!]

For a CSS code, what we need is $\partial_i \circ \partial_{i+1} = 0$, so we can draw on higher-dimensional chain complexes as well, and extending the complex further can be useful. Example: the 2D classical repetition code.

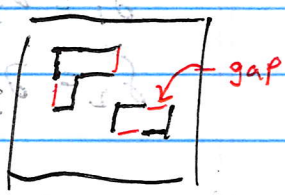
Bits are on plaquettes, checks on edges triggered if two plaquettes disagree:

$$0 \xrightarrow{\partial_3} V_2 \xrightarrow{\partial_2} V_1 \xrightarrow{\partial_1} V_0 \xrightarrow{\partial_0} 0$$



Now V_2 is a vector space assigning elements of $\mathbb{F}_2$ to each plaquette (1 indicates locating error). $\partial_2 = H$ is the parity check, assigning 1 to each edge where qubits disagree in $\partial_2 w$. $\partial_1 E$ assigns 1 to each vertex where # of edges in support meeting at vertex is odd

$\partial_2 \circ \partial_1 \Rightarrow$ the syndrome of this classical code is redundant — some syndromes cannot occur. That there is a constraint on the syndrome can be useful for syndrome decoding if there are syndrome measurement errors. There might be gaps in the sydrome, and we can guess how to fill them in.



This idea can also apply to a CSS code.

Example: 3D toric code.

Qubits live on plaquettes. Z stabilizers live on cubes (weight-6). X stabilizers live on edges

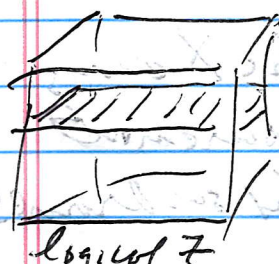
$$Z(c) = \prod_{p \in c} Z_p$$

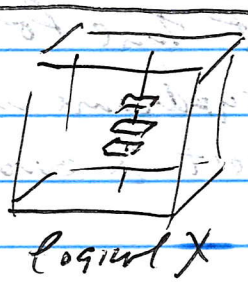
$$X(l) = \prod_{p \ni l} X_p$$

$0 \xrightarrow{\partial_4}$ cubes $\xrightarrow{\partial_3}$ plaquettes $\xrightarrow{\partial_2}$ links $\xrightarrow{\partial_1}$ sites $\xrightarrow{\partial_0} 0$

Z logicals are closed surface operators modulo trivial closed surfaces

X logicals are loop operators modulo trivial closed loops on the dual lattice





A stack of plaquettes winding around 3-torus.

Pairing of X-type and Z-type logicals

$K = 3$ logical qubits

X-type syndrome is a closed string of boundary of Z-type surface.

Chain complex ← commutes w. X stab

$$\mathcal{L}_Z = \frac{\ker \partial_2}{\mathrm{Im}\, \partial_3} = H_2(C)$$

↑ not in Z stab

$\delta_1 = \partial_2^T$ = fans out an edge to its plaquettes

$\delta_2 = \partial_3^T$ maps plaquette to its neighboring cubes

$$\mathcal{L}_X = \frac{\ker \delta_2 \leftarrow \text{commutes with Z stab}}{\mathrm{Im}\, \delta_1 \leftarrow \text{Not in X stab}} = H^2(C)$$

cubes ← plaquettes ← links ← sites

δ_2 δ_1 δ_0

The X-type stabilizers of Z errors are highly redundant $\partial_1(\text{syndrome 1-chain}) = 0 \Rightarrow$ syndromes form closed loops

This enables "single-shot" error correction for the Z errors. If there are measurement errors we can "fill in the gaps" if we have adequate confidence in the syndrome.

Unfortunately not true for the X errors. We need to repeat syndrome measurement multiple times to correct X errors if syndromes are noisy.