Lecture 21

Dec 3, 2025

Typically, when people talk about classical error correction they are talking about linear codes.

$k = \dim C$ and $C = \ker A$ ← check matrix.

Notation: $C = [n, k, d]$ code with locality ℓ if $C = \ker A$ with A being ℓ-row & ℓ-column sparse.

Ex. $A = \begin{pmatrix} 1 & 1 & 0 \\ 0 & 1 & 1 \end{pmatrix}$. $Ax = 0$ equals

$$x_1 \oplus x_2 = x_2 \oplus x_3 = 0.$$

$C = [3, 1, 3]$ code.

$$d = \min_{\substack{x \neq y \\ x, y \in C}} d_H(x, y) = \min_{\substack{x \in C \\ x \neq 0}} |x|.$$

Quantum Codes.

Let $C \subseteq (\mathbb{C}^2)^{\otimes n}$ be a Hilbert space s.t. $\dim C = 2^k$ and for all $E = E_S' \otimes \mathbb{1}_{[n] \setminus S}$ where $|S| < d$, we have

$$\langle \psi_1 | E | \psi_2 \rangle = 0 \quad \text{if} \quad |\psi_1\rangle \perp |\psi_2\rangle.$$

Equiv, dist d is the max d s.t. for all Paulis of size d,

$$\Pi P \Pi = \eta_P \Pi \quad \text{for } \Pi \text{ the projector onto } C.$$

Like prev, we can correct up to distances $\left\lfloor \frac{d-1}{2} \right\rfloor$.

Notation: $[[n, k, d]]$ code.

Shor's code is a $[[9, 1, 3]]$ code.

How do we build codes of better parameters?

To do, so we will study a special subclass of codes called Stabilizer codes due to their fundamental relation to Paulis and stabilizers.

Recall stabilizer states as the states defined by

linearly indep. and commuting Paulis $P_1, \ldots, P_n$.

Well if we only considered $P_1, \ldots, P_{n-k}$, there will be a 2^{n-k} dim subspace $\left(\cong (\mathbb{C}^2)^{\otimes k}\right)$ of states s.t.

$$P_i |\psi\rangle = |\psi\rangle.$$

Claim: $\mathrm{span}\{|000\rangle, |111\rangle\}$ is defined by Z_1Z_2, Z_2Z_3.

Pf. $Z_1Z_2 \left(\sum_x \alpha_x |x\rangle \right) = \sum_x (-1)^{x_1+x_2} \alpha_x |x\rangle$

so $\alpha_x = 0$ when $x_1 + x_2 = 1$.

Likewise, $\alpha_x = 0$ when $x_2 + x_3 = 1$. so

$\alpha_x \neq 0$ when $x \in \{000, 111\}$. ✓ □

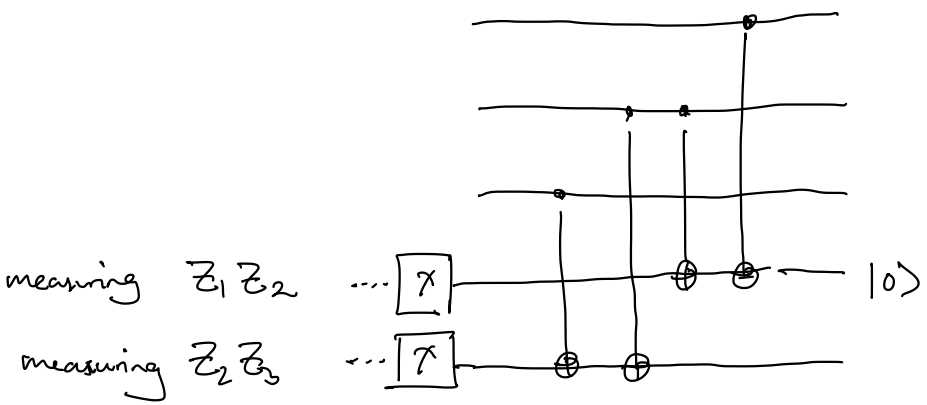
Recall notion of measuring w.r.t. an observable.

$$M = \Lambda_+ - \Lambda_-$$

↑ +1 eigenspace ↑ −1 eigenspaces, then

we have a POVM $\{\Lambda_+, \Lambda_-\}$.

The bit flip code has stabilizers Z_1Z_2, Z_2Z_3.
Where do these show up?



logical bit flip. $X_1X_2X_3 = \overline{X}$

commutes with all stabilizers.

Is there a logical phase flip? Yes. Z_1.

Since there is a 1 qubit phase flip, this cannot correct against phase flip errors.

Phase flip code stabilized by $X_1X_2,\ X_2X_3$.

logical phase flip. $Z_1Z_2Z_3$

logical bit flip. X_1.

What are the stabilizers of Shor code?

$$|i\rangle \mapsto \left(\frac{|0\rangle + (-1)^i |1\rangle}{\sqrt{2}}\right)^{\otimes 3} \mapsto \left(\frac{|000\rangle + (-1)^i |111\rangle}{\sqrt{2}}\right)^{\otimes 3}$$

$X_1X_2,\ X_2X_3$

$(Z_1Z_2,\ Z_2Z_3)$

$(Z_4Z_5,\ Z_5Z_6)$

$(Z_7Z_8,\ Z_8Z_9)$

$X_1X_2X_3\ X_4X_5X_6$

$X_4X_5X_6\ X_7X_8X_9.$

$\overline{X} = X_1 \cdots X_9, \quad \overline{Z} = Z_1 \cdots Z_9.$

Are there other logical bit and phase flips?

Table of stabilizers:

$$
\begin{aligned}
S_1 &= Z\ Z\ 1\ 1\ 1\ 1\ 1\ 1\ 1 \\
S_2 &= 1\ Z\ Z\ 1\ 1\ 1\ 1\ 1\ 1 \\
S_3 &= 1\ 1\ 1\ Z\ Z\ 1\ 1\ 1\ 1 \\
S_4 &= 1\ 1\ 1\ 1\ Z\ Z\ 1\ 1\ 1 \\
S_5 &= 1\ 1\ 1\ 1\ 1\ 1\ Z\ Z\ 1 \\
S_6 &= 1\ 1\ 1\ 1\ 1\ 1\ 1\ Z\ Z \\
S_7 &= X\ X\ X\ X\ X\ X\ 1\ 1\ 1 \\
S_8 &= 1\ 1\ 1\ X\ X\ X\ X\ X\ X \\
\bar{X} &= X\ X\ X\ X\ X\ X\ X\ X\ X \\
\bar{Z} &= Z\ Z\ Z\ Z\ Z\ Z\ Z\ Z\ Z .
\end{aligned}
$$

How do we correct and detect errors for stabilizer code? Suffices to consider Paulis.

Let C be stabilized by $\langle S_1, \ldots, S_{n-k} \rangle$

Three types of errors: Good, Bad, Ugly.

① Good error. E is a product of stabilizers. Then $E|\psi\rangle = |\psi\rangle$ and nothing changed.

② Bad error. E anticommutes with some S_i.

③ Ugly error. E commutes with all $S_1, \ldots, S_k$ but is outside their span.

Bad errors are detectable. To detect errors, measure each stabilizers S_i. If $ES_i = -S_i E$, then

$$S_i E|\psi\rangle = -ES_i|\psi\rangle = -E|\psi\rangle \quad \text{for } |\psi\rangle \in C,$$

Therefore S_i measurement outputs -1.

Ugly errors are logical transforms. They are undetectable as every stabilizer will measure $+1$ but the state changes.

Ex. For bit flip code

good	Z	Z	I
bad	I	I	X
ugly	X	X	X

Let $G = \langle S_1, \ldots, S_k \rangle$

The centralizer $C(G) = C_{P_n}(G)$ is the set

$$\{ P \in P_n \mid \forall g \in G,\ Pg = gP \}.$$

The set of Paulis which commutes with all of G.

Then the set of errors can be characterized by

$$\text{good} = G$$
$$\text{bad} = P_n \setminus C(G)$$
$$\text{ugly} = C(G) \setminus G.$$

A stabilizer code has distance d, if every Pauli error of size $< d$ is either good or bad

(equiv. trivial or correctable).

<u>Thm</u> For a stabilizer code on n-qubits with $n-k$ independent Pauli stabilizers $S_1, \ldots, S_{n-k}$, let $G = \langle S_1, \ldots, S_k \rangle$.

Then the rate of the code is k and the distance is the minimum size of a Pauli $\in C_{P_n}(G) \setminus G$.

Next: Kitaev's toric code. A construction of an error correcting code with local checks and distance growing with n.

Toric code is a special case of

Caulderbank-Shor-Steane (CSS) codes

where each stabilizer generator is either X-type or Z-type.

X-type $= X^a \leftarrow$ tensor product of only X terms.

Z-type $= Z^b \leftarrow$ tensor product of only Z terms.

Shor's code is also CSS.

Obs X-type checks detect for Z-errors and Z-type checks detect for X-errors.

only tensor product of Z and $\mathbb{1}$.

What is the smallest Z-error that is logical?

It's the smallest element of $C_{pn}(G) \setminus G$ which only consists of Z-terms.

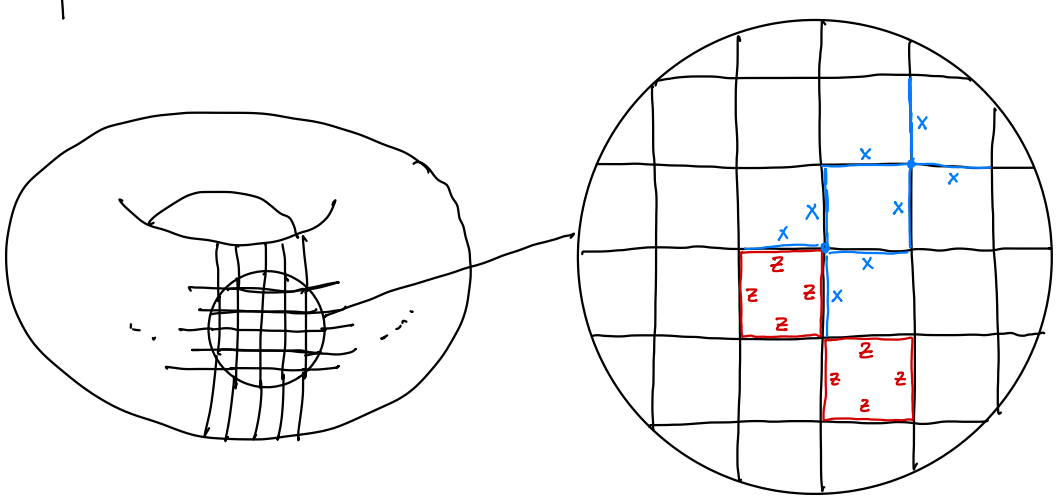
goal is to design a code s.t. all small Z-errors either are
(a) detected by the X-checks
(b) product of the Z-checks and (therefore, trivial as it acts like a stabilizer).

For $E \subseteq [n]$, let $Z_E = \mathbb{1} \otimes \mathbb{1} \otimes \ldots \underbrace{Z \otimes Z \otimes Z}_{\text{locations indicated by } E} \otimes \mathbb{1}$

An error Z_E is detected by a stabilizer X_A if $A \cdot E = 1$. Equiv., the size of the intersection $|A \cap E|$ is odd.

So, the "Z-distance" of the code is the smallest size error Z_E where intersection with every X-check X_A is even but Z_E is not a product of the Z-checks.

Place qubits on the edges of a grid-discretization of a torus.



For every face f, place a check Z_f which equals $\underbrace{Z \otimes Z \otimes Z \otimes Z}$ edges touching f.

And for every vertex v, place a check X_v which equals $\underbrace{X \otimes X \otimes X \otimes X}$ edges touching v.

All stabilizers commute as the intersection of a face f and a vertex v is either 0 or 2.

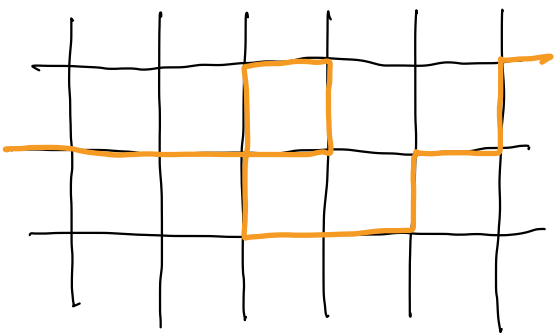
Two observations:

① A Z-error Z_E commutes with every X-check X_v iff E = union of cycles.

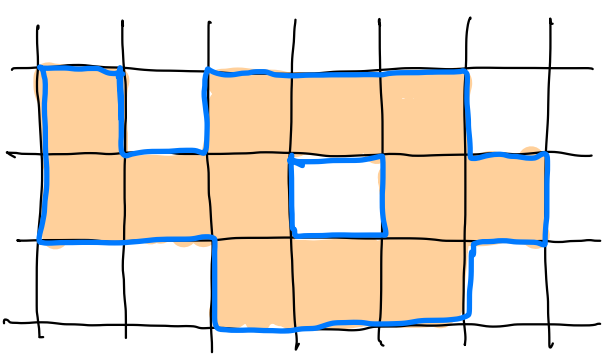
Pf. Use the edges $\in E$ to draw a graph (V, E).

The degree of every vertex is 0, 2, or 4. So the graph can be decomposed as a union of cycles.

"for every edge in, there is an edge out"



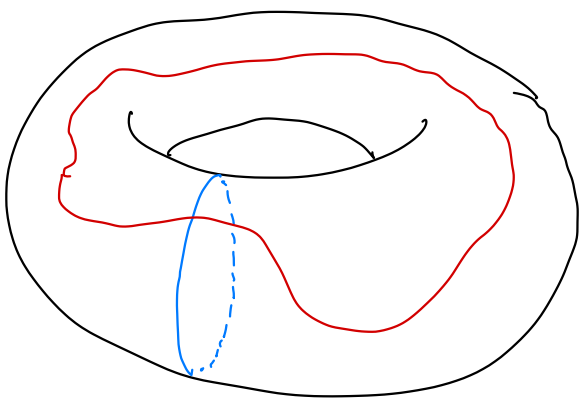
(2) The product of Z-checks is the boundary of a collection of faces.



What is the difference between cycles and boundaries?

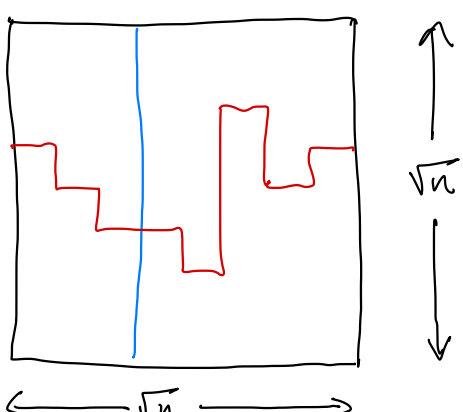
All boundaries are cycles but not all cycles are boundaries.

Example.



cycles which are not boundaries.

To see this its often easiest to unfold the torus.



Identify the top & bottom and identify the sides.

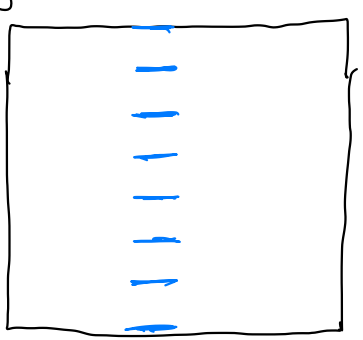
Cycles \ Boundaries = "non-trivial loops".

What is the shortest non-trivial loop?

Length = $\sqrt{n}$.

So, the Z-distance will be $\sqrt{n}$.

The X-distance is also $\sqrt{n}$ by a similar argument.



A non-trivial loop through the faces.

"co-cycles" \ "co-boundaries"

Correcting a general Pauli error.

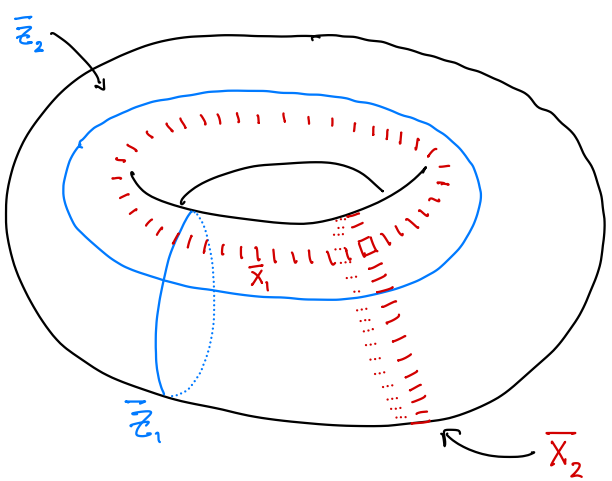
$$P = X_E Z_{E'}.$$

Since checks are only X- or Z-type, P anticommutes with X_v iff $|E' \cap v|$ is odd and with Z_f iff $|E \cap f|$ is odd.

Therefore, correctable if X_E and $Z_{E'}$ are both separately correctable.

What are the logical transformations for this code?

They will correspond to non-trivial loops.



Notice $\bar{X}_1$ and $\bar{Z}_1$ share an edge and therefore anticommute. Likewise $\bar{X}_2$ and $\bar{Z}_2$ anticommute. Other relations are commutation.

These logical operators are defined up to stabilizers.

By these relations, these define 2 logical qubits.

There are multiple pfs that these are the only logical qubits such as counting the number of independent stabilizers.

So, this is a $[[n, 2, \Omega(\sqrt{n})]]$ code.

For the longest time, this was the best known code. Today, we have constructions of $[[n, \Omega(n), \Omega(n)]]$ codes.

Lastly,

A rotated basis picture on stabilizer codes.

Let $G = \langle S_1, \dots, S_{n-k} \rangle$ for indep. stabilizers S_i.

Then $\exists$ unitary V s.t. $V S_i V^\dagger = Z_i$.

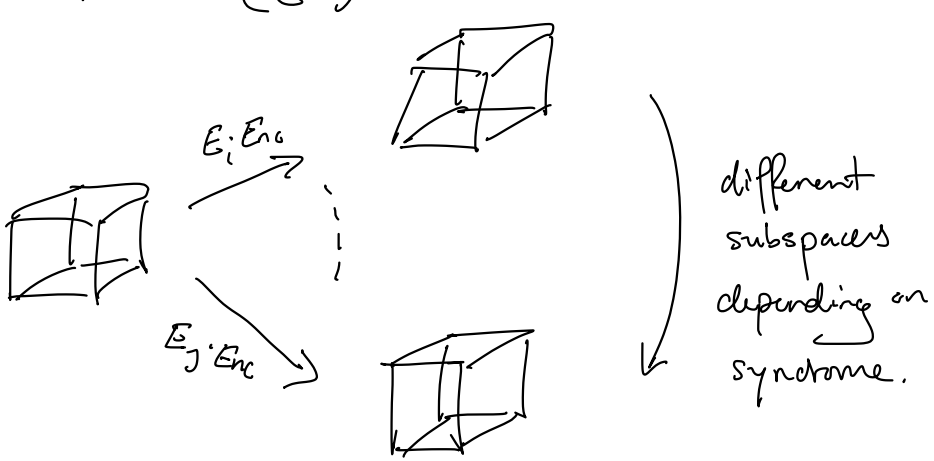
Then $VGV^\dagger = |0\rangle^{\otimes n-k} \otimes (\mathbb{C}^2)^{\otimes k}$

V is therefore the Encoding circuit.

Furthermore, if we measure the syndrome and get out $\vec{s} \in \{0,1\}^{n-k}$, then the state lies in

$$|\vec{s}\rangle \otimes (\mathbb{C}^2)^{\otimes n-k}.$$

So,



Today: Toric Code Correction & Hamiltonian Simulation.

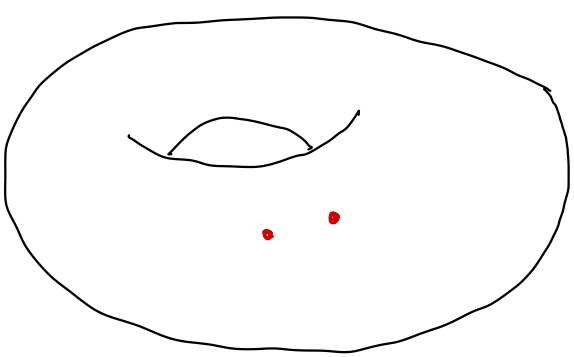
Correction: After measuring the syndrome (i.e. all the X-checks and Z-checks), we want to find the correction that takes us back to the codespace.

Classical repetition code:

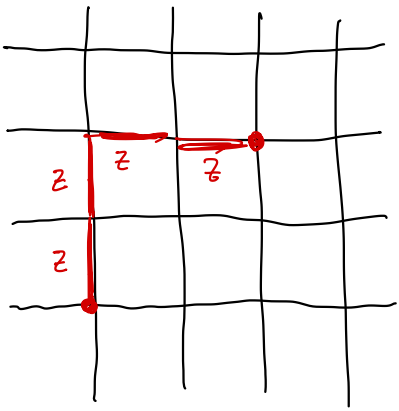
If most bits are 0, correct to 0^n.

If most bits are 1, correct to 1^n

First toric code example:



Assume syndrome measurements were $+1$ except at these two vertex checks.



X-checks correct

Z-errors.

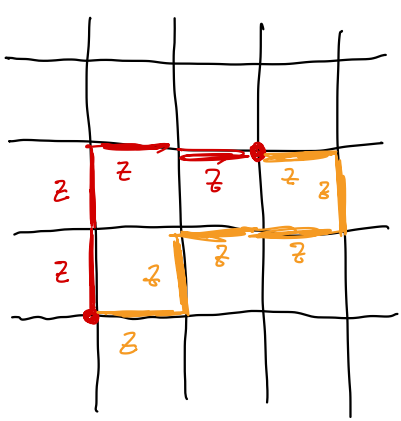
What Pauli errors can have this syndrome pattern?

Up to a trivial error, it must be a path connecting the two end points.

How do we know which path?

We don't. All paths are equivalent up to a trivial error.

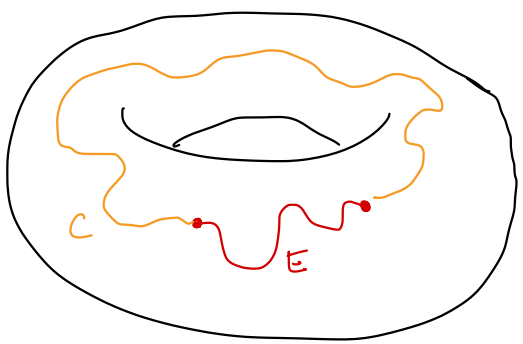
If the true error was in red, and the bad error in orange, correcting by flipping orange,



yields an overall trivial $\bar{Z}$-flip as it creates a boundary.

So, short answer is it didn't matter, as long as Error E + Correction C don't generate a non-trivial cycle.

Ex.

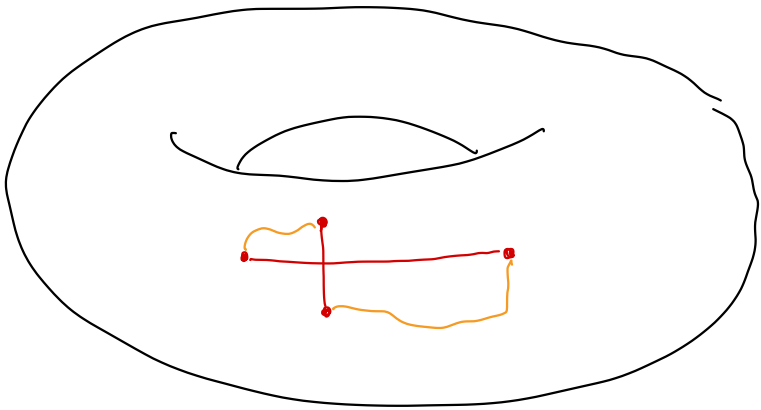


If our correction went around the torus, we would have a problem.

Correction assumes errors of size $< O(\sqrt{n})$, so we should only find corrections that are also short.

Note: computation of the correction can be done from the syndrome by a classical computer.

What if more than 2 vertices flag an error?



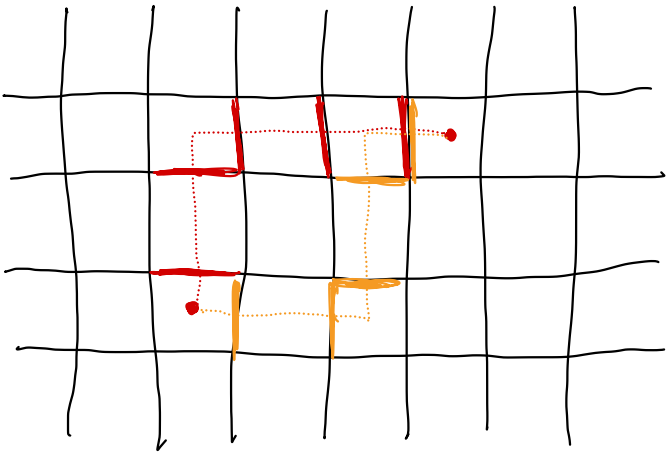
Just pair up flags in such a way that net correction length is $O(\sqrt{n})$.

If true error was red and correction orange, then error is still corrected.

Lastly, what about bit-flip errors and Z-checks flagging?

We look for co-path corrections.

These co-paths are equivalent up to co-boundary.



Hamiltonian Simulation

One of the strongest applications of q. computers is simulating q. mechanics which we believe would take exponential time on a classical computer.

Given a Hamiltonian $H = H(t)$, the evolution of a state is defined by

$$\frac{d}{dt}|\psi(t)\rangle = H(t)|\psi(t)\rangle.$$

When H is time-invariant,

$$|\psi(t)\rangle = e^{-iHt}|\psi(0)\rangle.$$

Matrix exponentiation: if $H = \sum_a \lambda_a |a\rangle\langle a|$ ← Spectral decomposition

then $e^{-iHt} = \sum_a e^{-i\lambda_a t} |a\rangle\langle a|$ ← unitary

Furthermore, $e^{-i(UHU^\dagger)t} = U e^{-iHt} U^\dagger$

Pf. $UHU^\dagger = \sum_a \lambda_a U|a\rangle\langle a| U^\dagger$

↑ new eigenvectors

$$e^{-i(UHU^\dagger)t} = \sum_a e^{-i\lambda_a t} U|a\rangle\langle a| U^\dagger$$

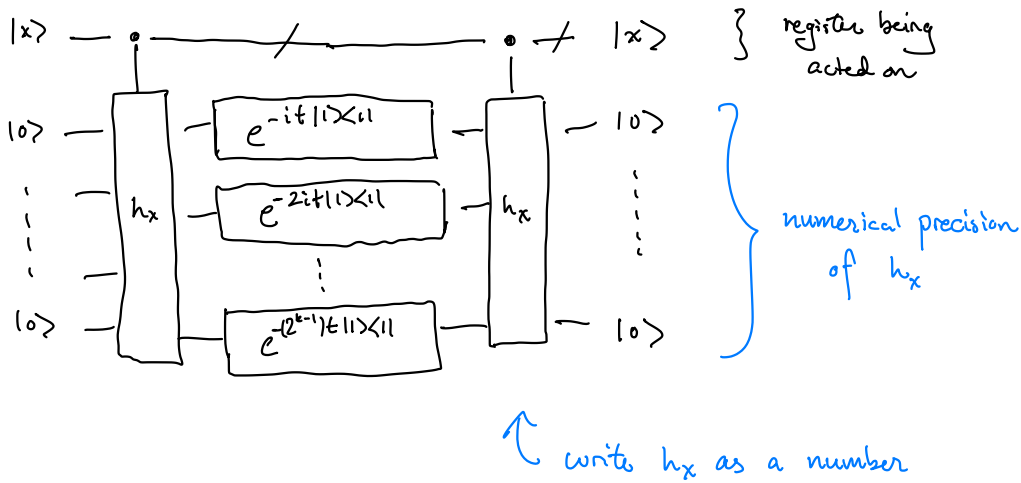
$$= U\left(\sum_a e^{-i\lambda_a t} |a\rangle\langle a|\right) U^\dagger. \quad \blacksquare$$

If we can diagonalize a Hamiltonian H s.t. $H = U H_{diag} U^\dagger$ then evolving w.r.t. H involves 2 applications of U + evolving by H_{diag}.

$$H_{diag} = \sum_x h_x |x\rangle\langle x| = \begin{pmatrix} h_0 & & & \\ & \ddots & & \\ & & & h_{2^\ell - 1} \end{pmatrix}$$

↑ real.

Circuit for $e^{-iH_{diag}t}$



If $h_x = h_x^{(0)} + 2h_x^{(1)} + 4h_x^{(2)} + \dots + (2^{k-1})h_x^{(k-1)}$

binary decomposition, then we map

$$|x\rangle \mapsto e^{-ih_x^{(0)}t} \cdot e^{-2ih_x^{(1)}t} \cdot \dots \cdot e^{-(2^{k-1})h_x^{(k-1)}t} |x\rangle$$

$$= e^{-ih_x t} |x\rangle$$

When H_{diag} is local, acting on a few qubits, this is implementable.

Furthermore, local Hamiltonian terms h_j can be diagonalized

easily. Therefore, we can implement the evolution of local Hamiltonian terms h_j easily.

What about the evolution of $H = \sum h_j$?

Even though each h_j is diagonalizable, the next sum may be hard to diagonalize and would end up a matrix on n-qubits

Solution:

$$e^{-i(h_1+h_2)t} = \lim_{m\to\infty} \left(e^{\frac{-ih_1 t}{m}} \; e^{\frac{-ih_2 t}{m}} \right)^m$$

Lie product formula.

$$\left\| \left(e^{\frac{-ih_1 t}{m}} \; e^{\frac{-ih_2 t}{m}} \right)^m - e^{-i(h_1+h_2)t} \right\| \leq \epsilon$$

for $m = O\left(\frac{(vt)^2}{\epsilon}\right)$ where $v = \max\left\{ \|h_1\|, \|h_2\| \right\}$.

Unappealing to evolve t^2 time for t-evolution, but there are speedups to get it to $t^{1+\delta}$ for any $\delta > 0$.

general evolution of $H = \sum_{j=1}^{J} h_j$. Let $m = O\left(\frac{(Jv\delta)^2}{\epsilon}\right)$

where $v = \max\{\|h_1\|, \ldots, \|h_J\|\}$ (typically 1 wlog).

① Compute circuit for $e^{-ih_j t'}$ for $t' = t/m$.

using diagonalization + diagonal evolution

② apply $\left(e^{-ih_1 t'} e^{-ih_2 t'} \cdots e^{-ih_J t'}\right)^m$.

Runtime $= \frac{Jm^2}{\epsilon}$ and can be improved to $\frac{Jm^{1+\delta}}{\epsilon}$.

Other simulable Hamiltonians include sparse Hamiltonians where H is mostly 0 except poly(n) entries per row.

∃ efficient alg for simulating evolution in this case.