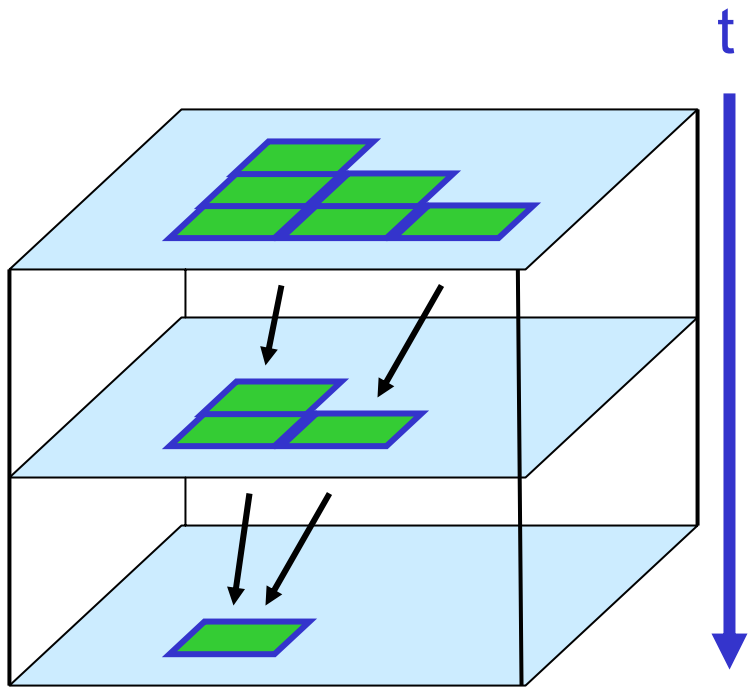
The cost of quantum fault tolerance



John Preskill, Caltech
14 January 2005
http://www.iqi.caltech.edu/

IQI Institute for Quantum Information
Sponsored by the NSF

Quantum Error Correction

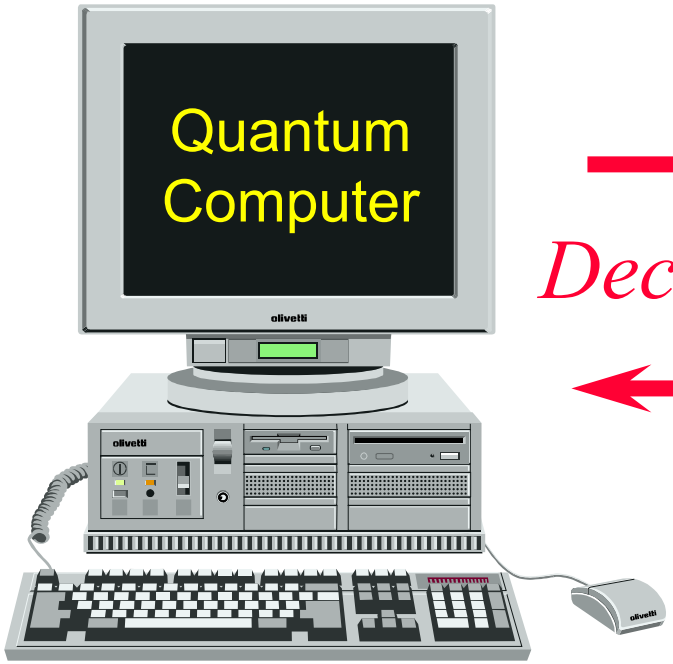
Shor ‘95

Steane ‘95

Quantum information can be protected, and processed fault-tolerantly.

Shor ‘95

Steane ‘95



Decoherence

Environment

ERROR!

If quantum information is cleverly encoded, it *can* be protected from decoherence and other potential sources of error. Intricate quantum systems *can* be accurately controlled.

Some recent developments in quantum fault tolerance

- Improved thresholds using error detection (Knill, Reichardt)
- Thresholds for non-Markovian noise (Terhal & Burkard)
- Thresholds for local gates (Svore, Terhal, DiVincenzo)
- Fault-tolerance with cluster states (Nielsen & Dawson)
- Purification of noisy ancillas (Bravyi & Kitaev)
- Nonabelian anyons from local interactions (Freedman, Nayak & Stengel, …)
- Topological storage in Josephson junction arrays (Ioffe et al.)
- Ion trap *experiments* (Wineland et al.)
- Etc.

Quantum fault tolerance

This subject is important, obviously, because quantum computers that solve hard problems will need to use fault-tolerant protocols.

It also connects with other deep issues in physics, e.g.: quantum order and quantum phase transitions, holographic encodings in quantum gravity, stable deformations of quantum mechanics, …

From a theorist's perspective, it is intriguing to compare fault-tolerance in the quantum world with fault tolerance in the classical world. There are strong similarities, but also important differences.

The cost of fault tolerance

A fundamental question about classical fault tolerance:

Suppose we are to simulate an ideal classical circuit with size *L* and depth *D* that computes a Boolean function, using noisy gates. Faults occur independently at the locations within the noisy circuit, where the probability of a fault at each location is no larger than ε (a constant). The circuit is to be executed with error probability δ, and "long wires" are allowed --- that is, we won't worry about how the bits are arranged in (e.g. three-dimensional) space. For what size and depth of the noisy circuit is this simulation possible?

In fact, for sufficiently small nonzero ε, there is an achievable simulation (for any Boolean function) with size and depth (Von Neumann '52, Dobrushin and Ortyakov '77, Pippenger '85, etc..)

$$L^* = O(L \log L) \quad D^* = O(D)$$

Furthermore, there are some Boolean functions (e.g. the parity of *n* bits) for which this size blow-up is necessary.

What is the answer to the corresponding question for quantum circuits?

The cost of quantum fault tolerance

Suppose we are to simulate an ideal quantum circuit with size L and depth D, using noisy gates. Faults occur independently in the noisy circuit, with fault probability no larger than ε. The probability distribution for the final measurement is to match the ideal probability distribution to constant accuracy δ. "Long wires" are allowed --- that is, we won't worry about how the qubits are arranged in (e.g. three-dimensional) space. For what size and depth of the noisy circuit is this simulation possible?

Using concatenated quantum codes, there is an achievable simulation (for sufficiently small ε) with size and depth (Aharonov & Ben-Or '96, Kitaev '96, etc..)

$$L^* = O(L \log^{\alpha} L) \quad D^* = O(D \log^{\beta} L)$$

(We assume that all gates are quantum gates and are included in our assessment of resources --- we don't allow fast accurate classical computation "for free.")

This depth blow-up is sizable, if the ideal circuit is highly parallelized. Can it be improved? Although we are considering an unphysical model in which gates are allowed to act nonlocally, this is a fundamental question about the distinction between classical and quantum fault-tolerance.

The cost of quantum fault tolerance

When concatenation is used, robustness is achieved via hierarchical organization, which causes the depth blow-up. Is there an alternative?

For our problem, *spatial* locality (e.g., in 3 dimensions) is not crucial, but to reduce circuit depth, it is desirable for each qubit to interact with only a small number of "neighbors."

We'll find it helpful to consider procedures that *are* spatially local in *d* dimensions. Then as *d* increases, these procedures become more robust (e.g., each qubit has more neighbors and there is more redundancy in the error syndrome).

Using topological codes in four or more spatial dimensions, and a local recovery algorithm, the depth blow-up can be improved to (Ahn & JP)

$$L^* = O(L \log^{\gamma} L) \quad D^* = O(D \log\log L)$$

Open question: Can the depth blow-up be reduced further to a constant?

The cost of quantum fault tolerance

- Size and depth blow-up for classical repetition coding, and for quantum concatenated codes.

- Toom's rule: local recovery operation for the two-dimensional classical repetition code.

- Toric quantum code in two dimensions.

- Toom's rule for the toric quantum code in four dimensions.

- Achievability of quantum size and depth blow-ups:

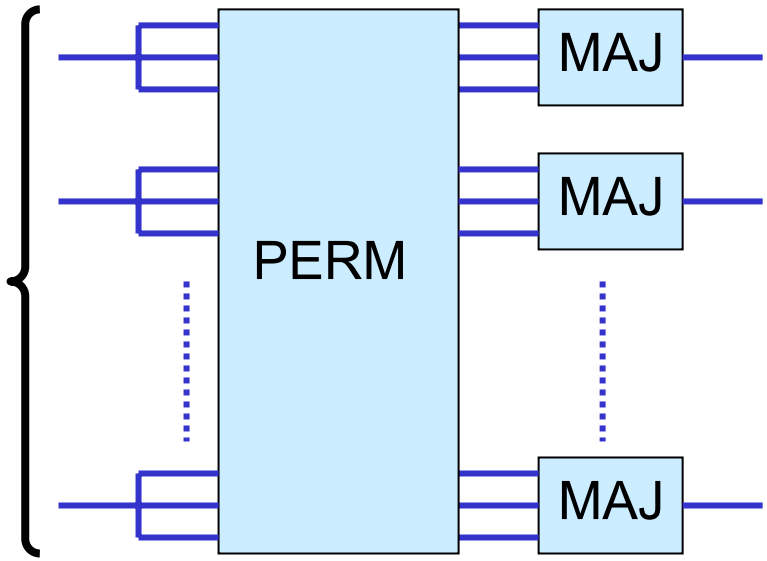
$$L^* = O(L \log^{\gamma} L) \quad D^* = O(D \log\log L)$$

The cost of fault tolerance

This classical recovery circuit uses the *n*-bit repetition code. There exists positive $\alpha < ½$ and positive β such that:

If $< \alpha n$ input errors and $< \beta n$ *faults*, then $< \alpha n$ output errors.

A gate gadget (gate followed by recovery) has a similar property. Therefore, if each gadget has fewer than βn faults, then the repetition code can always be decoded successfully.



The simulation of a gate has a constant depth. The probability of a gate failure is

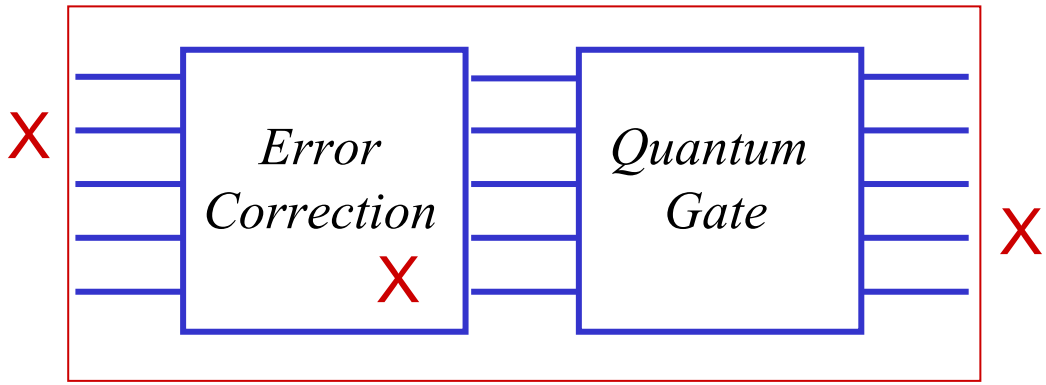
$$P_{\text{fail}} = O\left(\varepsilon^{\beta n}\right) < \delta / L \Rightarrow \beta n = O\left(\frac{\log L/\delta}{\log 1/\varepsilon}\right)$$

Therefore:

$$L^* = O(L \log L) \quad D^* = O(D)$$

Fault-tolerant quantum gates

For a distance-5 code (which can correct two errors), construct a "gate rectangle" consisting of a fault-tolerant gate preceded by fault-tolerant error correction on each input block. Then:



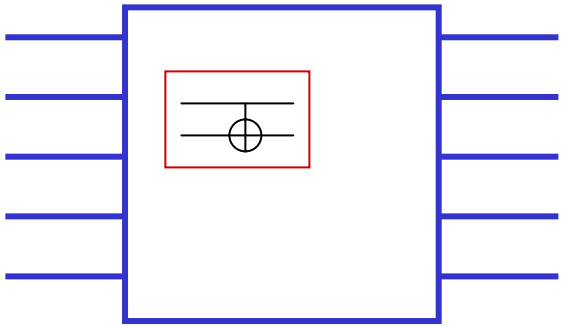
If the input has at most one error in each block, and the gate rectangle contains at most one fault, then the output has at most one error in each block.

Two faults in one rectangle:

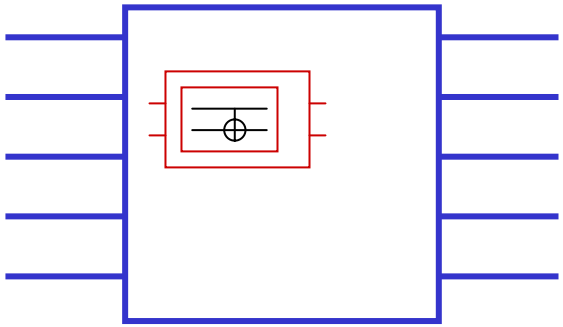
$$P_{\text{fail}} \leq L A_{\max} \varepsilon^2$$

where $A_{\max}$ is an upper bound on the number of pairs of circuit locations in each rectangle. Therefore, by using a quantum code that corrects two errors and fault-tolerant quantum gates, we can improve the circuit size that can be simulated reliably to $L=O(\varepsilon^{-2})$, compared to $L=O(\varepsilon^{-1})$ for an unprotected quantum circuit.

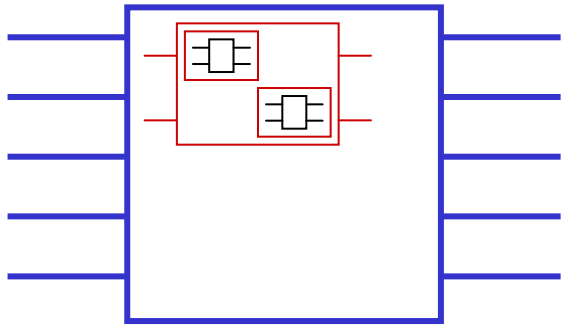
Recursive simulation



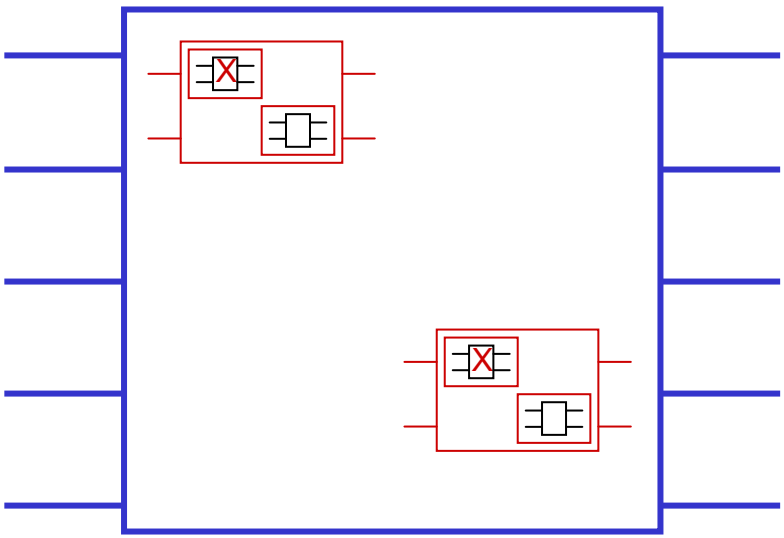
A *1-rectangle* is built from quantum gates.



A *2-rectangle* is built from 1-rectangles.



A *3-rectangle* is built from 2-rectangles.



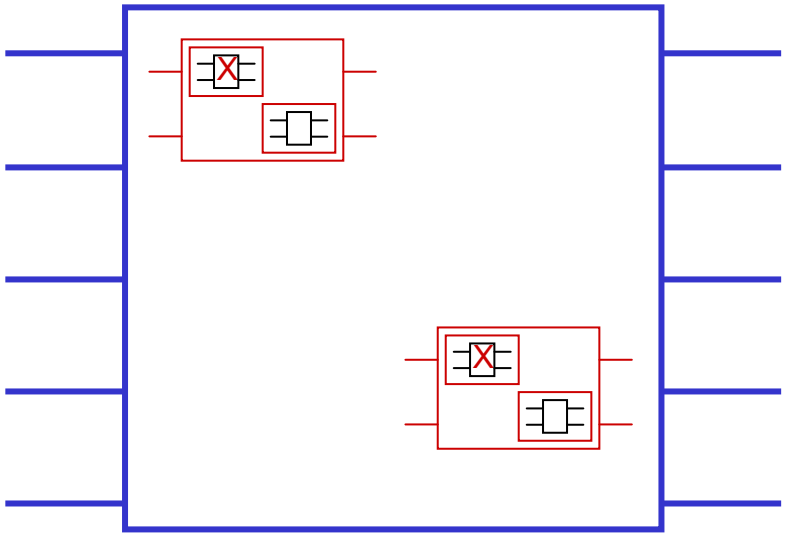
Definition:

A 1-rectangle is *good* if it contains no more than one fault; otherwise it is *bad*.

An r-rectangle is *good* if it contains no more than one bad (r-1)-rectangle; otherwise it is *bad*.

A level-r recursive simulation will be successful if every r-rectangle is good.

Recursive simulation



A 1-rectangle is *good* if it contains no more than one fault; otherwise it is *bad*.

An r-rectangle is *good* if it contains no more than one bad (r-1)-rectangle; otherwise it is *bad*.

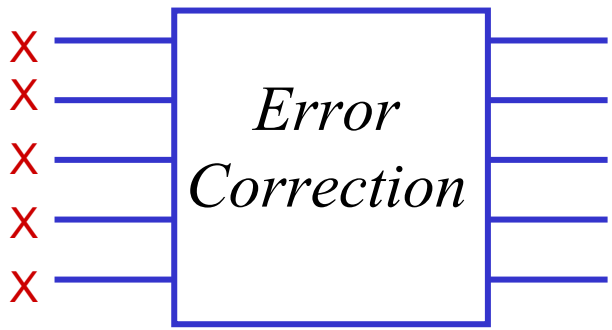
A 1-block is *correct* if it contains no more than one error; otherwise it is *incorrect*.

An r-block is *correct* if it contains no more than one incorrect (r-1)-block; otherwise it is *incorrect*.

Lemma: If the input to a good r - rectangle is correct, then the output is correct.

To prove this, we need another property of our error correction procedure:

For any input and no faults, the output is a *codeword*.



The level-r circuit simulation succeeds if all r-rectangles are good.

Recursive simulation

Lemma: If the input to a good r - rectangle is correct, then the output is correct.

The level-r circuit simulation succeeds if all r-rectangles are good. How likely is a bad r-rectangle?

$$P_{\text{bad}}^{(1)} \le A\varepsilon^2 \; ; \quad P_{\text{bad}}^{(r)} \le A\left(P_{\text{bad}}^{(r-1)}\right)^2$$

$$\Rightarrow P_{\text{bad}}^{(r)} \le A^{-1}\left(A\varepsilon\right)^{2^r}$$ (*double* exponential scaling in r)

Therefore, for a circuit with L locations (including "identity gates")

$$P_{\text{fail}}^{(r)} \le LA^{-1}\left(A\varepsilon\right)^{2^r} < \delta$$

is satisfied provided that $\varepsilon < \varepsilon_0 = A^{-1}$ and

$$\left(\varepsilon/\varepsilon_0\right)^{2^r} < \delta/\varepsilon_0 L \quad \Rightarrow \quad 2^r > \frac{\log\left(\varepsilon_0 L/\delta\right)}{\log\left(\varepsilon_0/\varepsilon\right)}$$

Recursive simulation

Suppose 1-rectangles have block size *n*, depth *d*, and size a. Then *r*-rectangles have block size n^r, depth d^r, and size a^r. Therefore:

$$(\varepsilon/\varepsilon_0)^{2^r} = \delta/\varepsilon_0 L \quad \Rightarrow \quad 2^r = \frac{\log(\varepsilon_0 L/\delta)}{\log(\varepsilon_0/\varepsilon)} \quad \Rightarrow$$

$$a^r = 2^{r\log_2 a} > \left[\frac{\log(\varepsilon_0 L/\delta)}{\log(\varepsilon_0/\varepsilon)}\right]^{\log_2 a} \qquad d^r = 2^{r\log_2 d} > \left[\frac{\log(\varepsilon_0 L/\delta)}{\log(\varepsilon_0/\varepsilon)}\right]^{\log_2 d}$$

Quantum Threshold Theorem: (Aharonov & Ben-Or, Kitaev, etc.) Suppose that faults occur independently at the locations within a quantum circuit, where the probability of a fault at each location is no larger than ε. Then there exists $\varepsilon_0 > 0$ such that for a fixed $\varepsilon < \varepsilon_0$ and fixed $\delta > 0$, any circuit of size L and depth D can be simulated by a circuit of size L^* and depth D^* with accuracy greater than δ, where, for constants $\alpha = \log a$ and $\beta = \log d$,

$$L^* = O\left[L(\log L)^{\alpha}\right] \qquad D^* = O\left[D(\log L)^{\beta}\right]$$

Recursive simulation

Quantum Threshold Theorem: (Aharonov & Ben-Or, Kitaev, etc.) Suppose that faults occur independently at the locations within a quantum circuit, where the probability of a fault at each location is no larger than ε. Then there exists $\varepsilon_0 > 0$ such that for a fixed $\varepsilon < \varepsilon_0$ and fixed $\delta > 0$, any circuit of size L and depth D can be simulated by a circuit of size L^* and depth D^* with accuracy greater than δ, where, for constants $\alpha = \log a$ and $\beta = \log d$,

$$L^* = O\left[L (\log L)^{\alpha} \right] \qquad D^* = O\left[D (\log L)^{\beta} \right]$$

Similar scaling of the overhead can also be achieved by concatenating a distance-3 code, which can correct one error. In that case, we need to carry out the inductive step for *overlapping rectangles* (Knill, Laflamme & Zurek; Aliferis, Gottesman & JP).

The numerical value of the *accuracy threshold* ε_0 is of practical interest --- for the above scaling of size and depth, we know that $\varepsilon_0 > 10^{-4}$ … With a higher overhead cost, the threshold can be larger (Knill, Reichardt, etc.).

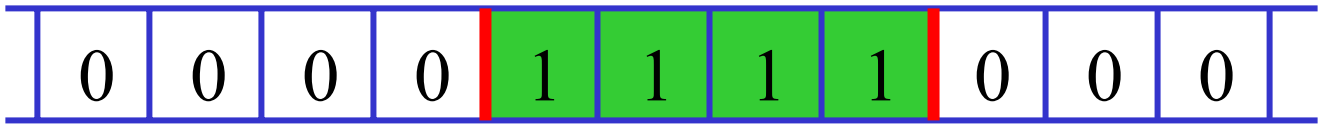
Improving the depth blow-up

- When concatenation is used, robustness is achieved via hierarchical organization, which requires a depth blow-up. Is there an alternative?

- For our problem, *spatial* locality (e.g., in 3 dimensions) is not crucial, but to reduce circuit depth, it is desirable for each qubit to interact with only a small number of "neighbors."

- We'll find it helpful to consider procedures that *are* spatially local in d dimensions. Then as d increases, these procedures become more robust (e.g., there is more redundancy in the error syndrome).

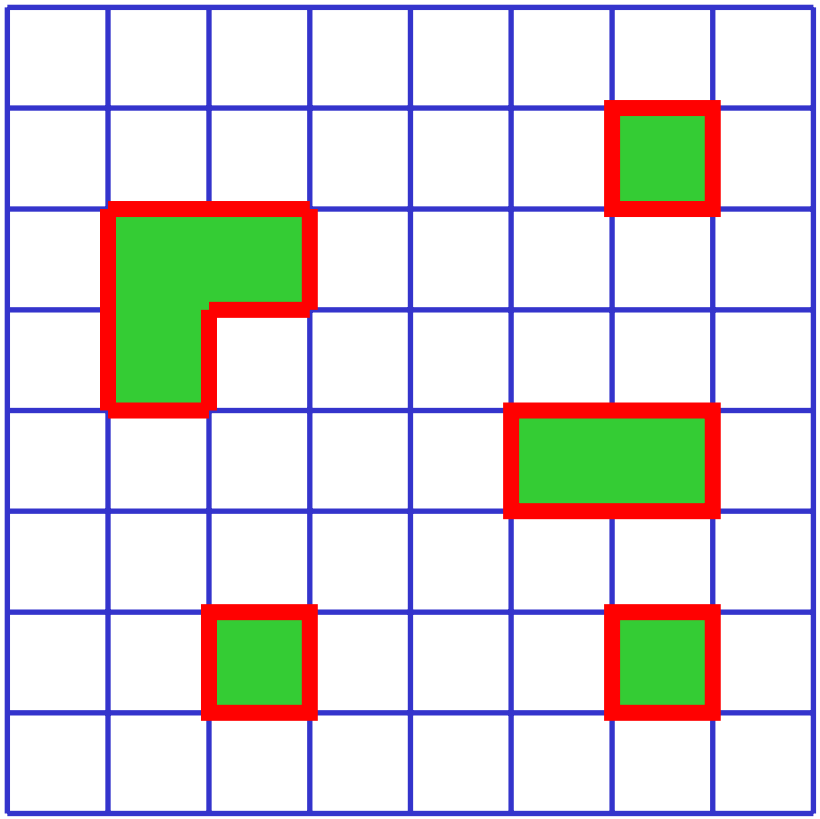
Example: 1D Ising model (repetition code)



When a connected (one-dimensional) droplet of flipped spins arises due to a thermal fluctuation, only the (zero-dimensional) boundary of the droplet contributes to the energy; thus the energy cost is independent of the size of the droplet.

Therefore, thermal fluctuations disorder the spins at any nonzero termperature. A one-dimensional ferromagnet is not a robust (classical) memory.

2D Ising model (repetition code)



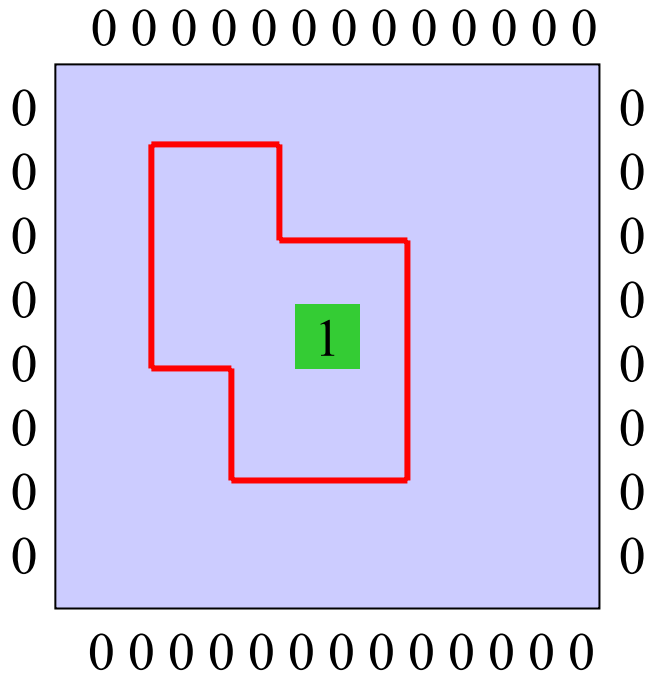
This memory is a repetition code, but with redundant (hence robust) parity checks.

Again, droplets of flipped spins arise as thermal fluctuations. But now the energy cost of a (two-dimensional) droplet is proportional to the length of its (one-dimensional) boundary.

Therefore, droplets with linear size L are suppressed at sufficiently low nonzero temperature T by the Boltzman factor $\exp(-L / T)$, and are rare.

The probability of a memory error becomes exponentially small when the block size is large. (Actual storage media, which are robust at room temperature, rely on this physical principle.)

Peierls (1936) argument for two Gibbs states



Suppose that all spins at the boundary point up. In thermal equilibrium, what is the probability that the spin at the origin points down?

The origin is contained in a droplet of flipped spins, and there must be a domain wall (of length l) that encloses the origin…Then if the temperature is sufficiently small:

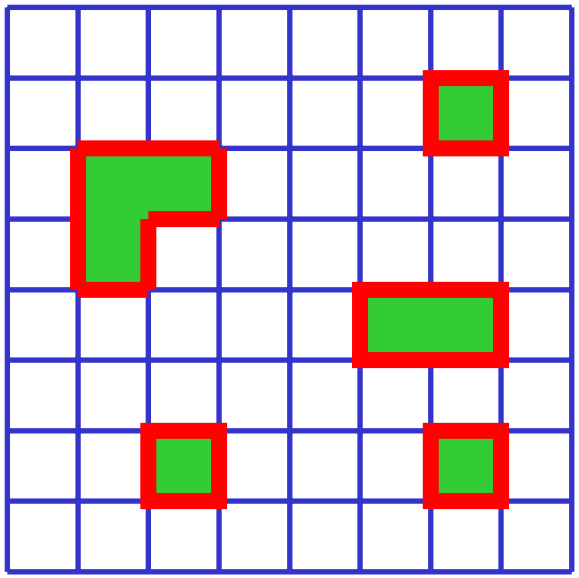
$$\text{Prob}[1 \text{ at origin}] \leq \sum_{l} l^2 3^l \exp\left[-l/T\right] < 1/2$$

Bounds number of paths

Bounds prob of each path

Similarly, if the spins at the boundary point down, the spin at the origin prefers to point down. In the limit of an infinite system, and at sufficiently low temperature, there are two Gibbs states in the Ising model.

Toom's (1980) rule: robust memory in a 2D PCA



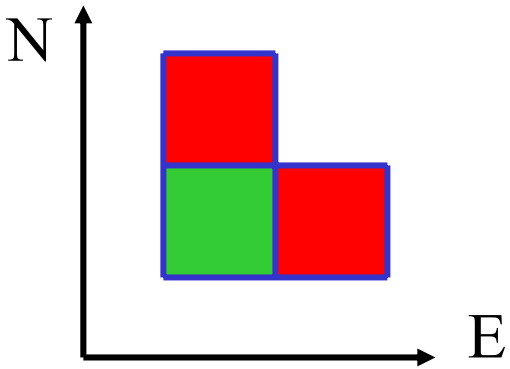
Drawing inspiration from the 2D Ising model and the Peierls argument, Toom (1980) showed that a two-dimensional probabilistic (i.e., noisy) cellular automaton can function as a robust memory: there are two translation-invariant stationary probability distributions for the states of the cells.

When applied noiselessly, Toom's (anisotropic) deterministic local rule removes a droplet of flipped spins in a time proportional to the linear size of the droplet ("eroder condition").

Noise causes droplets to appear, and slows the pace of droplet removal. But if the noise is weak, a typical droplet wil be removed in "linear time,", and large droplets are exponentially rare, as in the Ising model.

Furthermore, while a nonzero magnetic field destabilizes one of the Gibbs states of the Ising model, Toom's memory is robust even for biased noise.

Toom's rule: robust memory in a 2D PCA

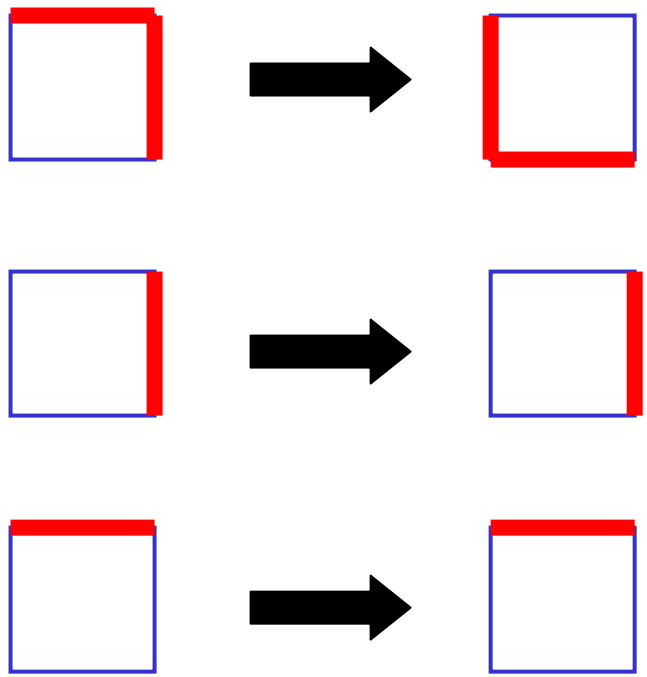


In each time step, each cell updates itself by taking a majority vote of itself and its north and east neighbors.

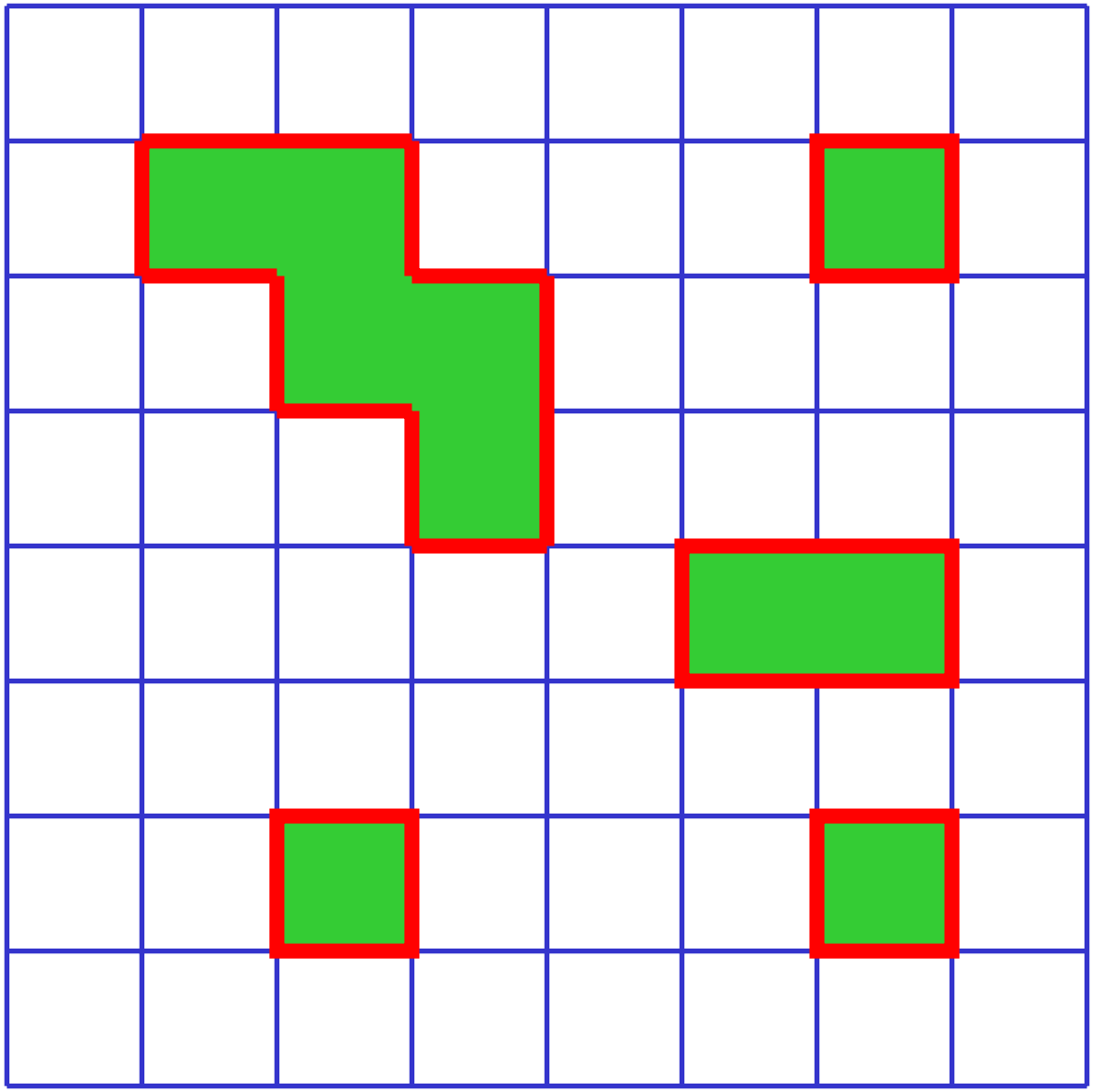
Equivalently, the cell examines its north and east edges, and flips itself if there is a domain wall ("string") on both edges.

Toom's rule causes the northwest boundary of a droplet to migrate toward the southwest.

The anisotropy of the rule is important --- it prevents confusion about which way the domain wall should move.



Toom's rule in the 2D PCA

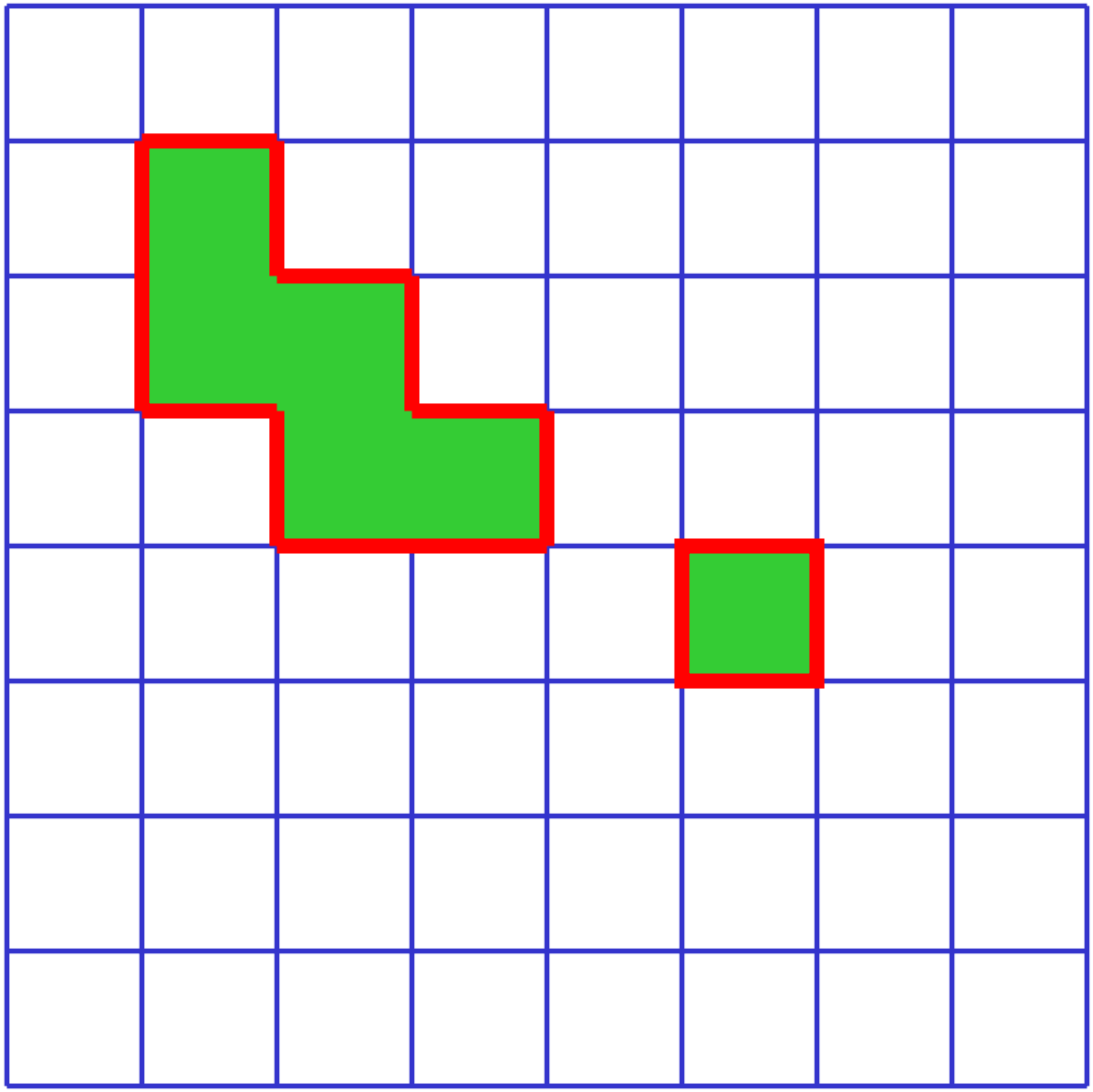


An anisotropic deterministic local rule causes droplets to shrink:

A cell flips if there is a wall on both its north and east edge.

Thus, Toom's rule causes the northeast boundary of a droplet to migrate toward the southwest.

Toom's rule in the 2D PCA

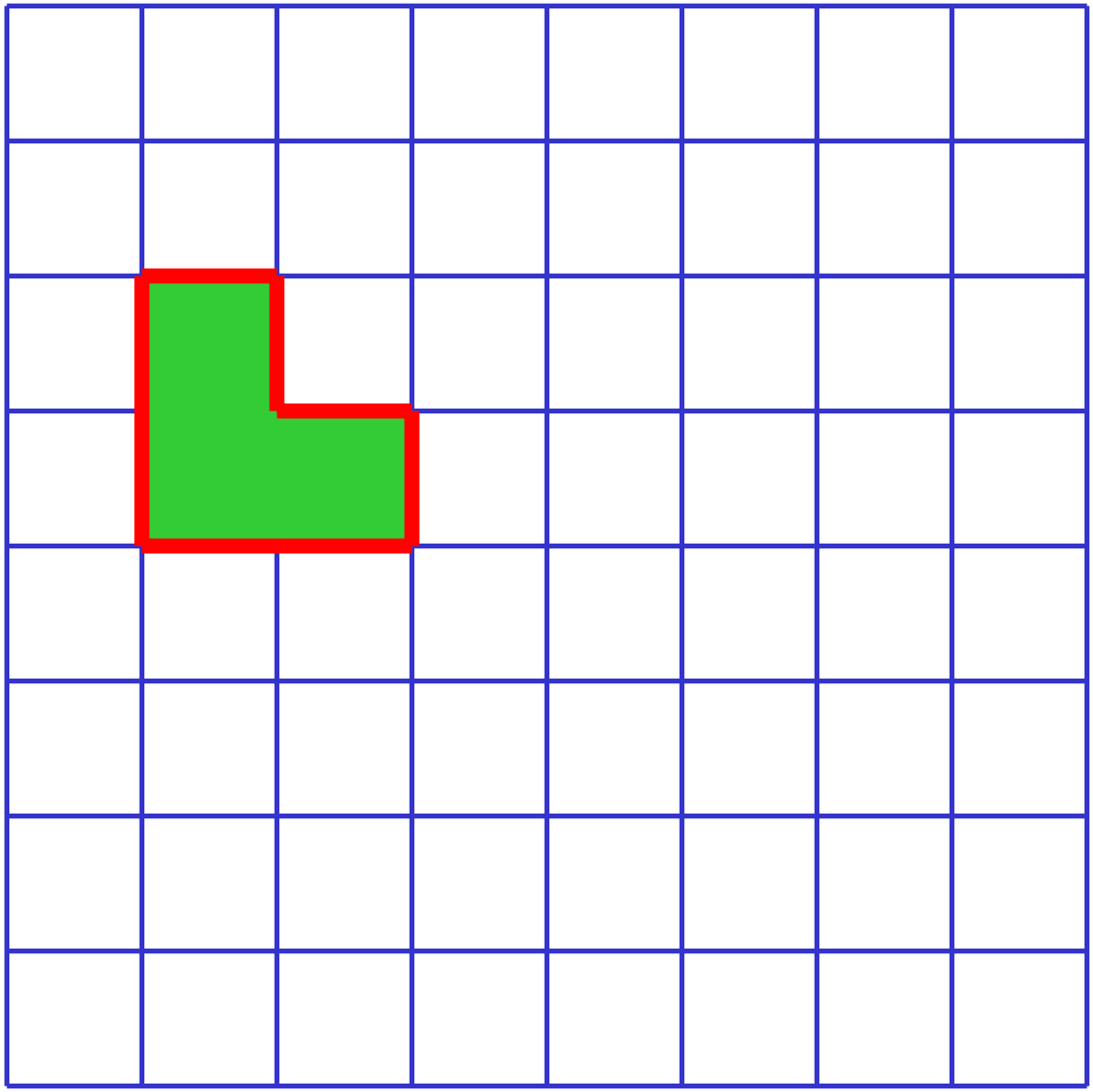


An anisotropic deterministic local rule causes droplets to shrink:

A cell flips if there is a wall on both its north and east edge.

Thus, Toom's rule causes the northeast boundary of a droplet to migrate toward the southwest.

Toom's rule in the 2D PCA

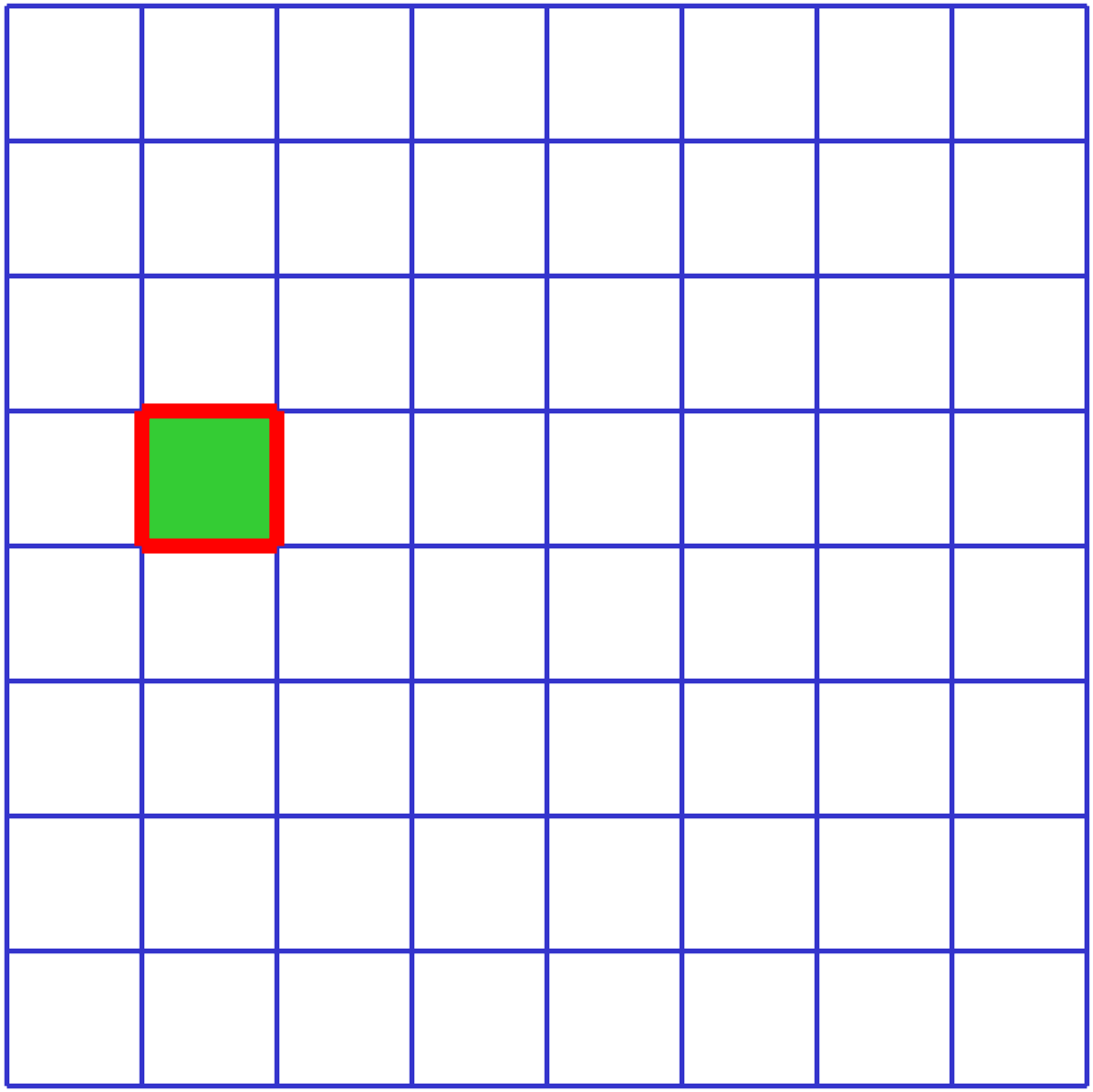


An anisotropic deterministic local rule causes droplets to shrink:

A cell flips if there is a wall on both its north and east edge.

Thus, Toom's rule causes the northeast boundary of a droplet to migrate toward the southwest.

Toom's rule in the 2D PCA

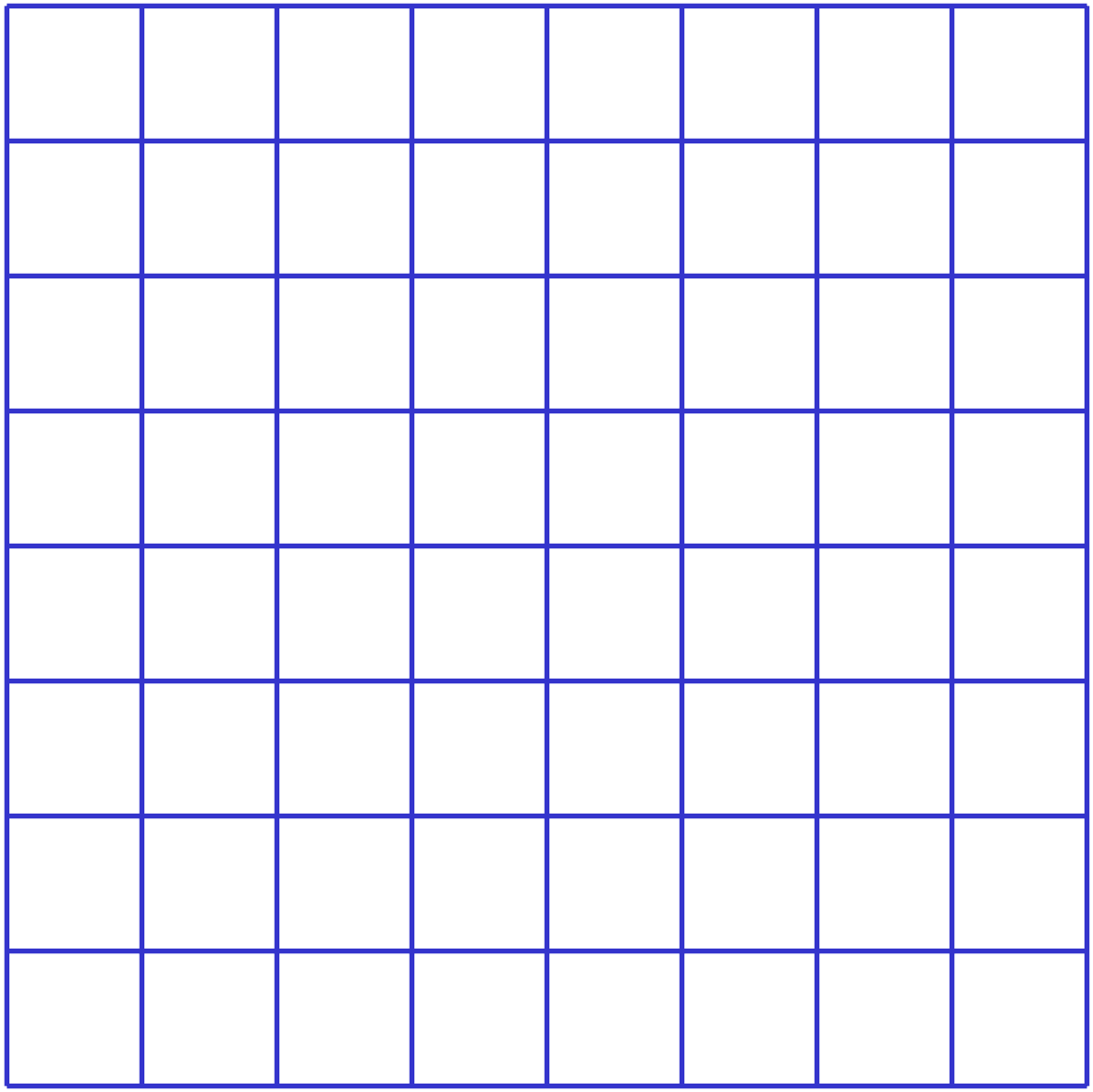


An anisotropic deterministic local rule causes droplets to shrink:

A cell flips if there is a wall on both its north and east edge.

Thus, Toom's rule causes the northeast boundary of a droplet to migrate toward the southwest.

Toom's rule in the 2D PCA



An anisotropic deterministic local rule causes droplets to shrink:

A cell flips if there is a wall on both its north and east edge.

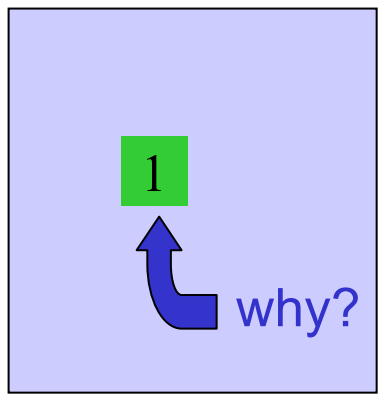
Thus, Toom's rule causes the northeast boundary of a droplet to migrate toward the southwest.

“Peierls argument” for Toom’s rule

Toom’s rule “sees” the boundary of a droplet (and erodes the droplets northeastern edge) much as the Hamiltonian of the Ising model sees the boundary. We can anticipate that if the fault rate is small, droplets are typically small and dilute, and most cells are correct.This intuition can be vindicated by a counting argument.

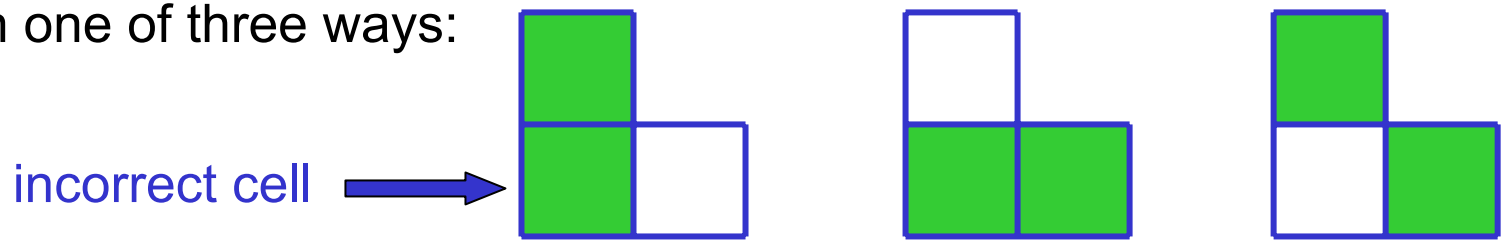
To bound the probability that a particular cell is incorrect, Toom (also Berman & Simon, Gács) uses a more sophisticated version of the Peierls argument, in which graphs in spacetime are counted (rather than domain wall contours at a fixed time). Each graph describes the locations in spacetime of faults that explain why the cell in question is incorrect.

An explanation is *minimal*, in that if any of the faults were absent, the cell would be correct. We can obtain an upper bound on the probability that the cell is incorrect by summing the probabilities of all these minimal explanations.



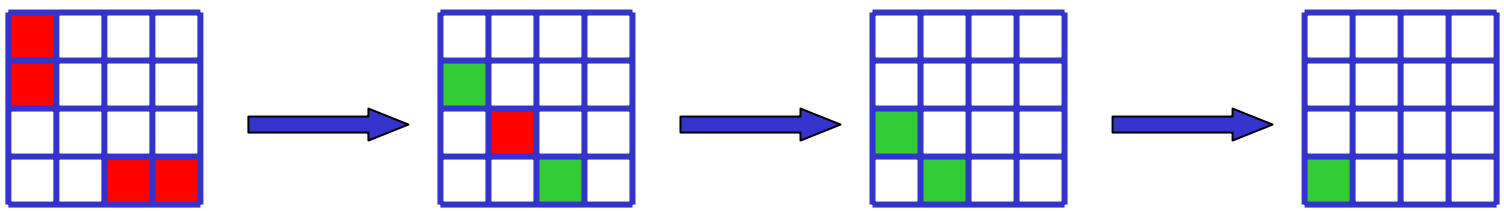
“Peierls argument” for Toom’s rule

For example, an incorrect cell at time 0 could be explained by a fault at time 0; otherwise, there must be two incorrect cells at time -1 that explain it, chosen in one of three ways:



In turn, these two incorrect cells could be explained by two faults at time -1, or by two faults at time -2 and another fault at time -1, or …

Example:

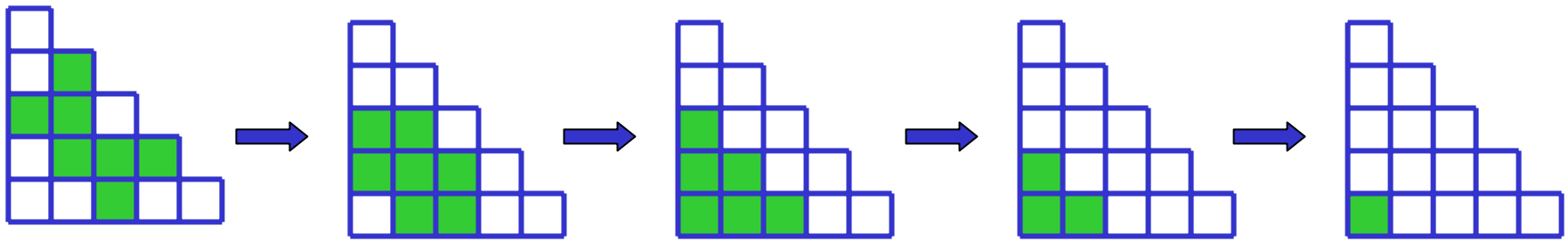


In this case, two droplets arise from faults (shown in red) at time -3; these erode, and are joined together by another fault to form a larger droplet at time -2, which erodes to a single cell at time 0. If the fault rate is ε, the probability of this scenario is $< \varepsilon^5$.

We wish to count such histories.

“Peierls argument” for Toom’s rule

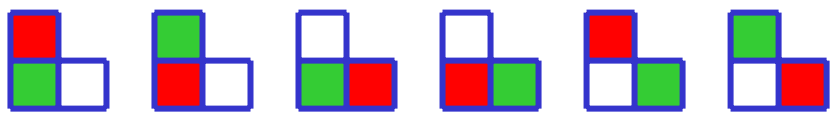
A “connected” droplet has a “size,” the number of successive applications of Toom’s rule that are needed to remove the droplet. This size is the side length of an Isosceles right triangle that contains the droplet:



The droplet has “poles” --- cells on its western, southern, and northeastern edges.

A *minimal* explanation is characterized by all events in which droplets form due to faults, or in which the *poles* of pre-existing droplets are joined by new faults. For an explanation with n faults, there are (n-1) events in which faults join droplets, and at most (n-1) steps in which droplets erode.

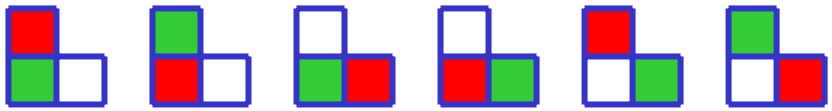
There are six ways for a fault to join a droplet:

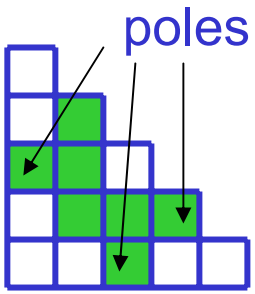


“Peierls argument” for Toom’s rule

A *minimal* explanation is characterized by all events in which droplets form due to faults, or in which the *poles* of pre-existing droplets are joined by new faults. For an explanation with n faults, there are (n-1) events in which faults join droplets, and at most (n-1) steps in which droplets erode.

There are six ways for a fault to join a droplet:

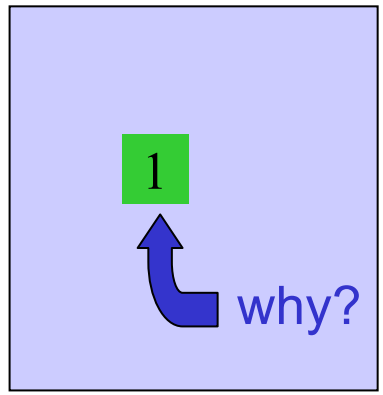




For an eroding droplet, each pole at time t is descended from a pole at time (t-1), which can be chosen in any of three ways.

Therefore, a minimal explanation of an incorrect cell caused by n faults can be characterized by a graph that is (essentially) a tree. The root is the incorrect cell, the leaves are the n faults, other nodes are poles of eroding droplets, and there are at most 4(n-1) edges. The tree is drawn on a graph with maximal node degree 6+6=12 (6 for the ways that a fault can join a droplet, and 6 for the ways to choose the antecedent and descendent of a pole).

"Peierls argument" for Toom's rule



Explanation tree: the root is the incorrect cell, the leaves are the n faults, other nodes are poles of eroding droplets, and there are at most 4(n-1) edges. The tree is drawn on a graph with maximal node degree r = 6+3=12 (6 for the ways that a fault can join a droplet, and 6 for the ways to choose the antecedent and descendent of a pole).

number of explanations with n faults

$$\text{Prob[1 at origin]} \le \sum_{n=1}^{\infty} |E_n| \, \varepsilon^n$$

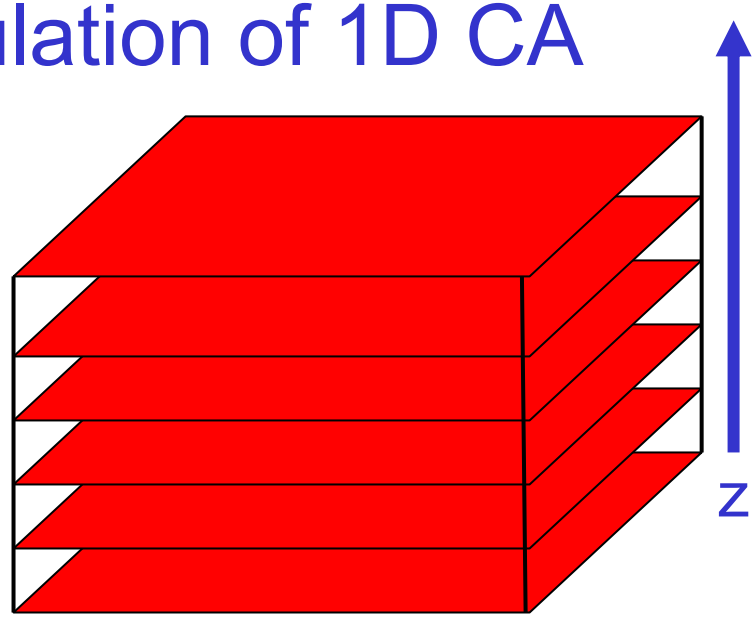
There is an accuracy threshold, because growth of the number $|E_n|$ of explanations is exponential (as for the domain wall contours of Peierls).

$$|E_n| \le A(B\varepsilon)^n \quad \Rightarrow \quad \varepsilon_{\text{th}} \ge (2B)^{-1} = \left(2r^2\right)^{-4} \Big/ 2$$

[A far from optimal threshold estimate: from numerics (Bennett, …) the actual threshold is about 5%.]

Toom's rule: robust 3D simulation of 1D CA

By stacking the two-dimensional Toom blocks in the third dimension, we can achieve a robust simulation of a 1D cellular automaton (Gács & Reif): Toom's rule is executed in the horizontal planes in between successive vertical applications of the 1D CA rule.

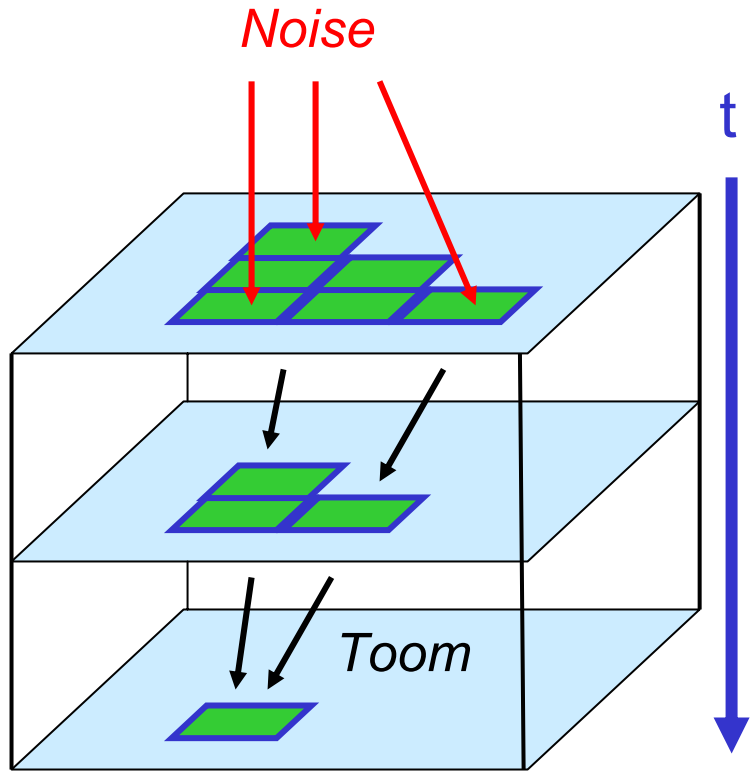


Now we need to worry about *propagation of error* from each slice to the neighboring slices. But the analysis is essentially the same. The only difference is that now the explanation tree is drawn on a graph of larger maximum degree, because an incorrect cell at time t and height z can be explained by a pair of incorrect cells at time (t-1), where the members of this pair might be on the neighboring slices at height (z+1) or (z-1). This propagation depresses the value of the threshold by a modest amount.

[Aside: combining an anisotropic local rule and hierarchical organization, a robust local simulation is possible even in 1D (Gács), but the proof is *much* more complicated.]

Toom's rule: suppression of large droplets

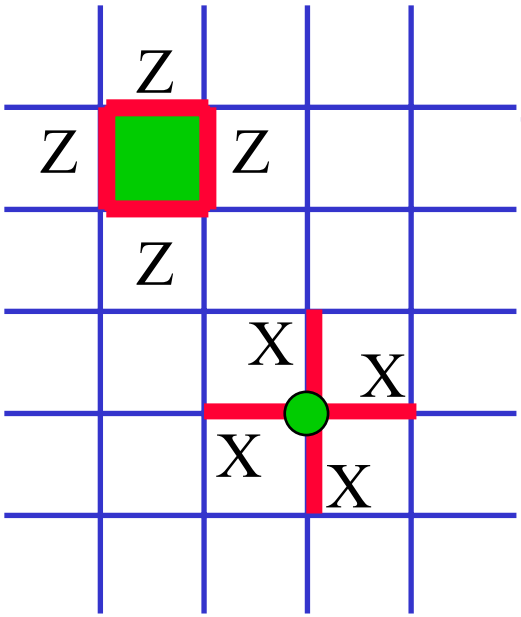
At each fixed time, how rare are connected droplets of size *s*? The droplet is reduced to an isolated single incorrect cell after *s* iterations of the *noiseless* Toom rule. Therefore, each explanation of the droplet is an explanation of this single incorrect cell, with at least *n* = *s* faults.



$$\text{Prob[droblet of size s]} \le \sum_{n=s}^{\infty} |E_n| \, \varepsilon^n = O\left(\varepsilon^s\right)$$

Large droplets are exponentially suppressed (as in the Ising model).

Toric code (Kitaev '96)



Qubits are arranged at the edges of a square lattice on a two-dimensional surface (e.g., the torus), and the check operators are four-qubit operators that can be measured *locally*:

$$\bigotimes_{+} X = 1, \qquad \bigotimes_{\square} Z = 1,$$

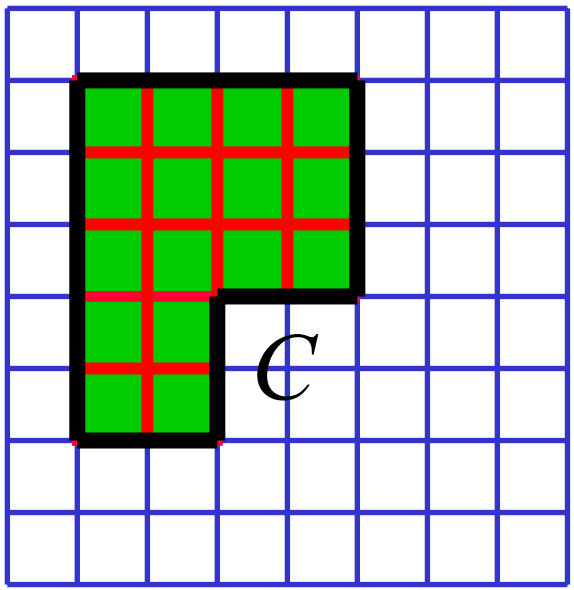
$$X = \begin{pmatrix} 0 & 1 \\ 1 & 0 \end{pmatrix} \qquad Z = \begin{pmatrix} 1 & 0 \\ 0 & -1 \end{pmatrix}$$

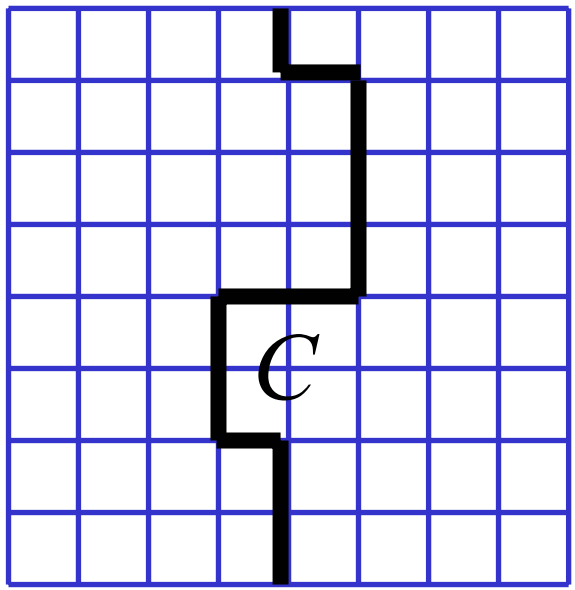
These operators are mutually commuting (and so can be simultaneously diagonalized). The code space is their simultaneous eigenspace.

The X operators, which detect Z errors, are defined on lattice sites; the Z operators, which detect X errors, are defined on lattice plaquettes (sites of the dual lattice).

On the torus: The code space $\mathfrak{H}_{\text{code}}$ (all check ops =1)

is preserved by $\bigotimes_C Z$ if *C* is a *cycle* (has no boundary).

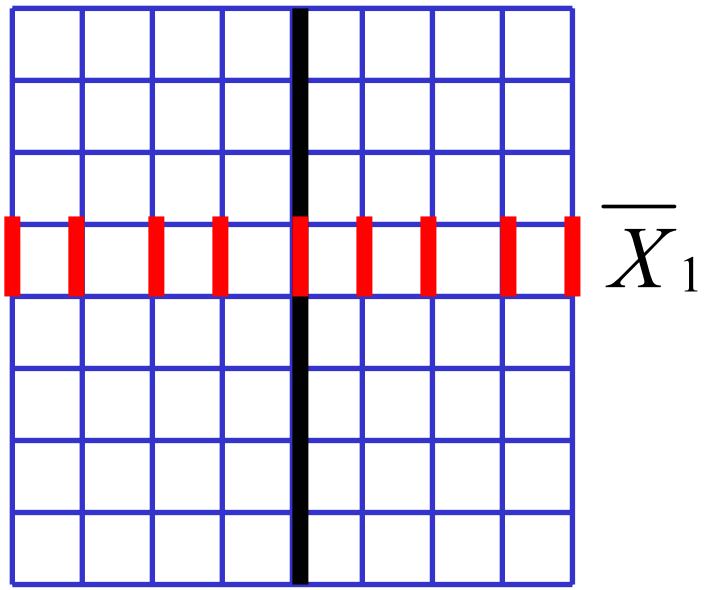


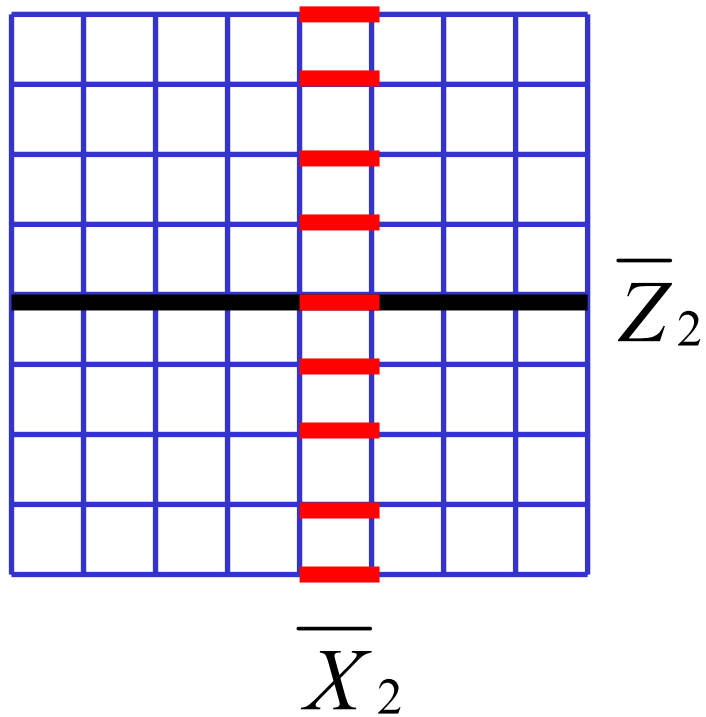


Homologically trivial; $\bigotimes_C Z$ is a product of check operators. It acts trivially on the code space.

Homologically nontrivial; $\bigotimes_C Z$ is *not* a product of check operators.It is a logical operation acting on the code subspace.

Two Encoded Qubits



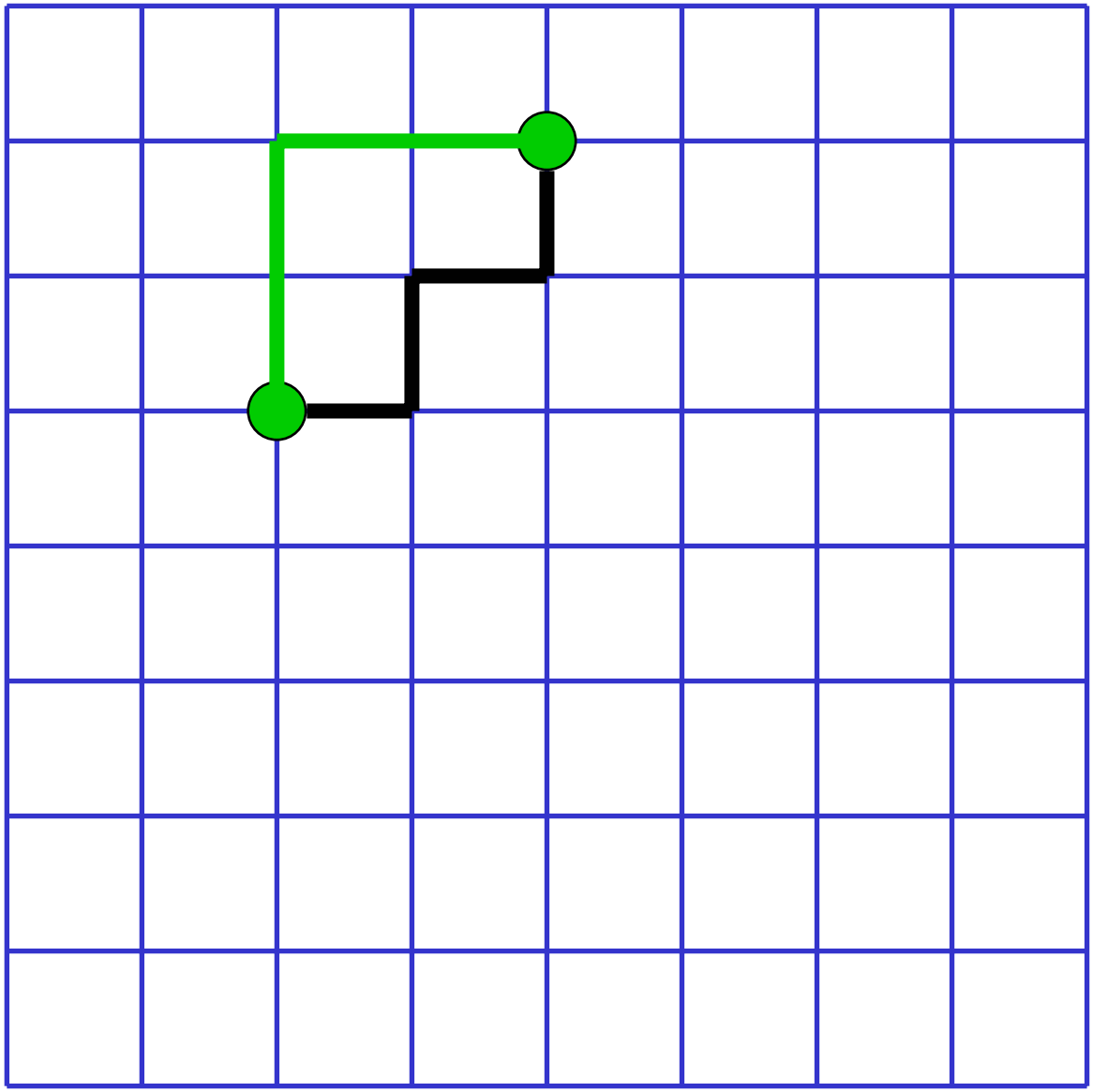


$\overline{Z}_{1,2}$: nontrivial cycles of lattice

$\overline{X}_{1,2}$: nontrivial cycles of dual lattice

The quantum information is encoded *nonlocally*. In contrast to the classical repetition code, it is not locally accessible.

Toric Code Recovery

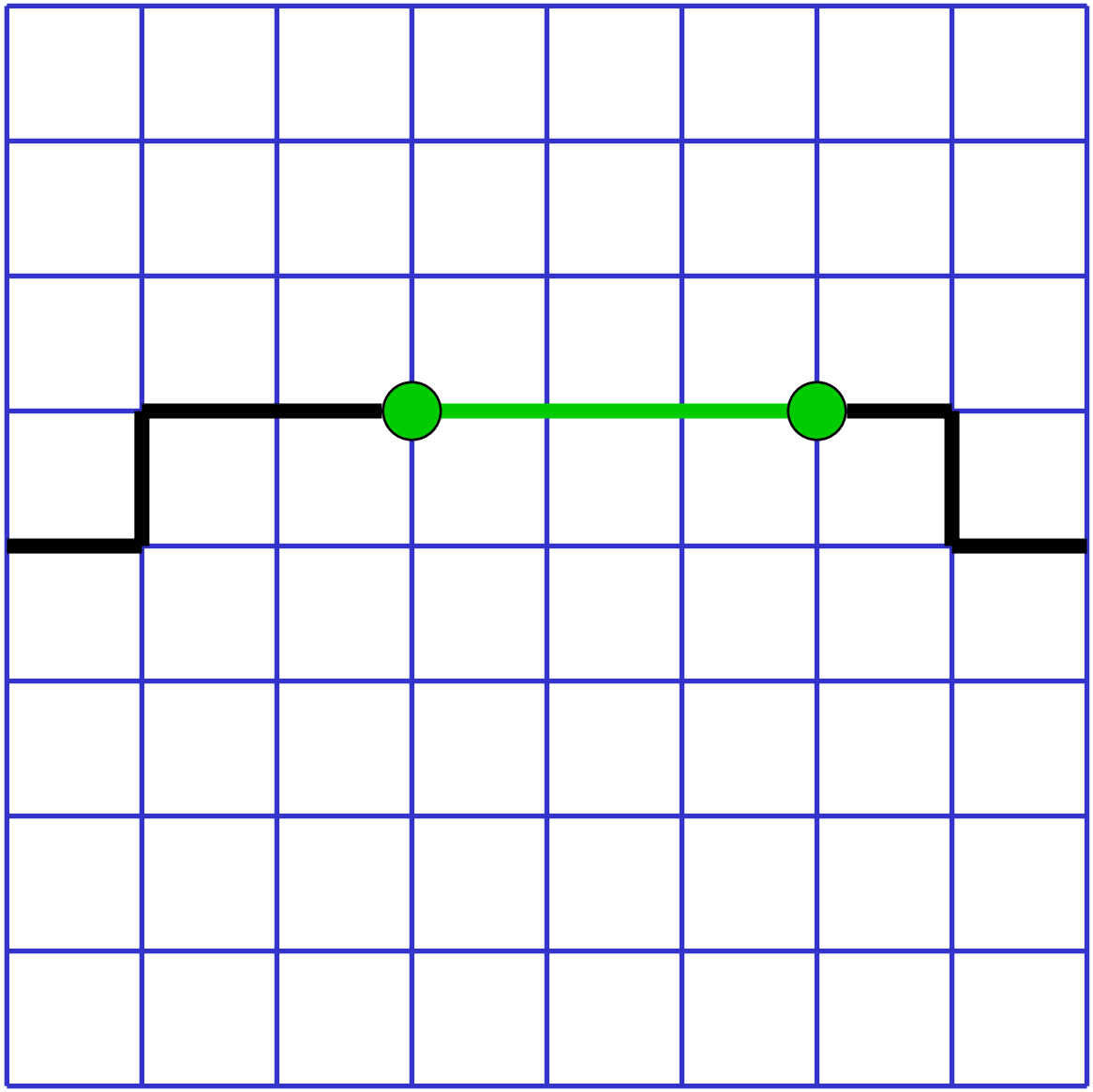


Measuring the syndrome detects the endpoints of a chain of errors.

The syndrome is highly ambiguous; many different error chains can have the same boundary.

But as long as our interpretation of the syndrome is *homologically* correct, recovery will be successful.

Toric Code Recovery

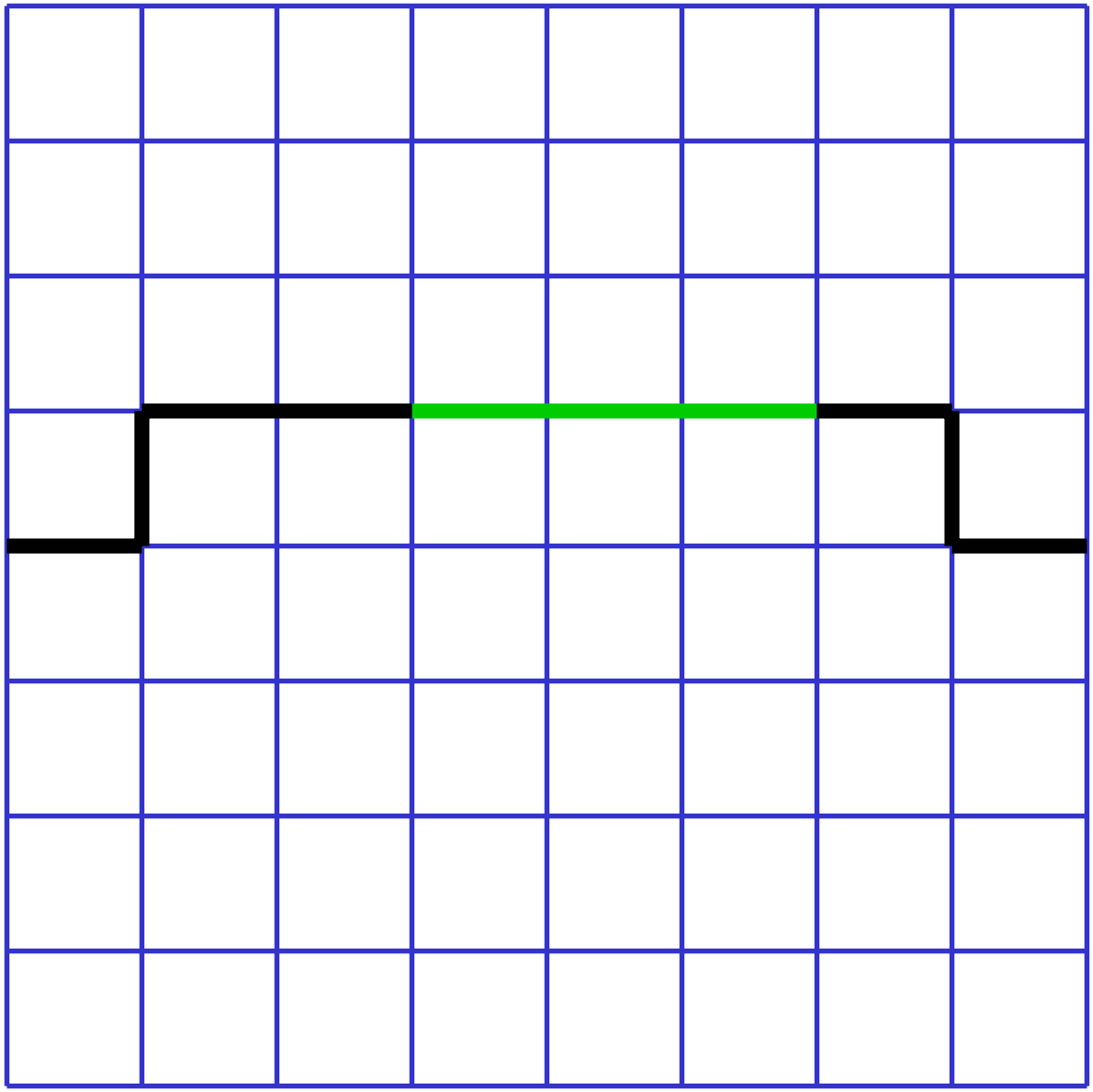


Measuring the syndrome detects the endpoints of a chain of errors.

The syndrome is highly ambiguous; many different error chains can have the same boundary.

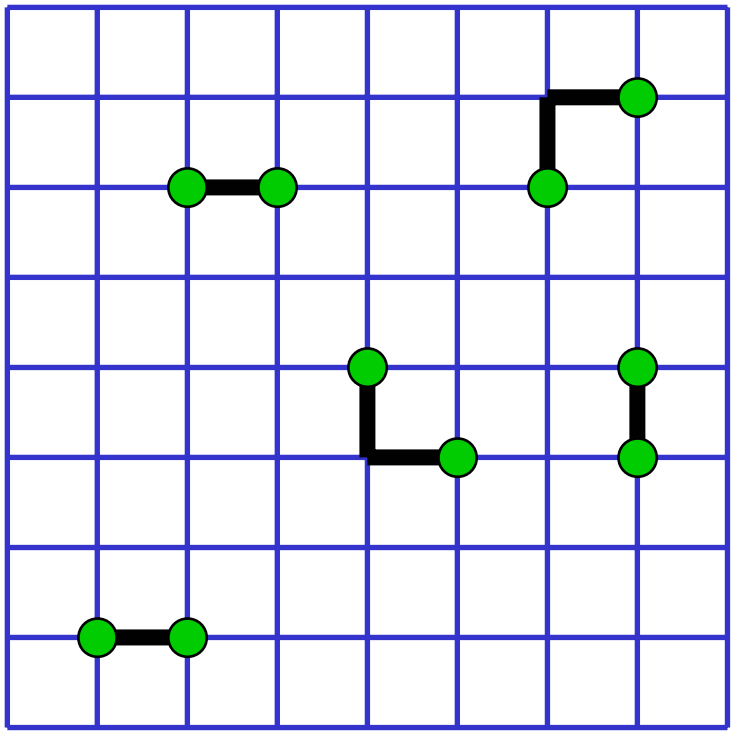
But as long as our interpretation of the syndrome is *homologically* correct, recovery will be successful.

Toric Code Recovery



ERROR!

Toric Code Recovery



If the error rate is small, the chain segments are typically short, and the defect positions are strongly correlated. Therefore, it should be "easy" to guess how to pair the defects. We should be able to tolerate a number of defects that scales linearly with the block size (even though the distance of the code increases as the square root of the block size).

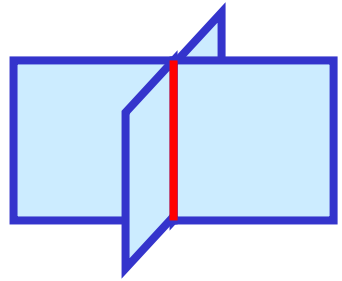
There is an accuracy threshold.

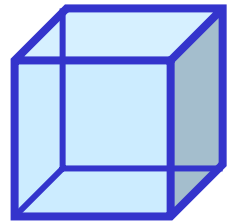
The probability of an encoded error is $e^{-O(L)}$ on an L X L torus, if we assume accurate and instantaneous (poly-time) classical processing, and if in each round, the probability of a qubit error and of a syndrome measurement error are both below 3%. Recovery succeeds if the failure rate of the local quantum gates used for syndrome measurement is below $\varepsilon_0 > 1.7 \times 10^{-4}$ (Dennis, Kitaev, Landahl & JP).

Toric code in *four* dimensions

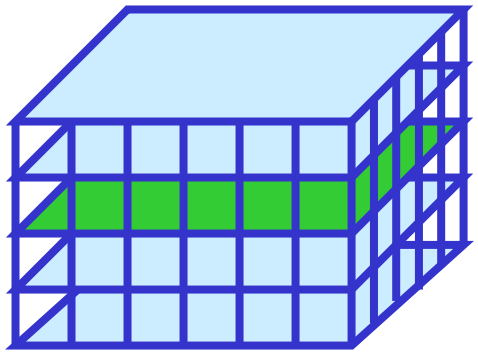
-- Qubits are on *plaquettes* (2-cells):

-- 6-qubit X and Z check operators at *edges* and *cubes* (dual links):

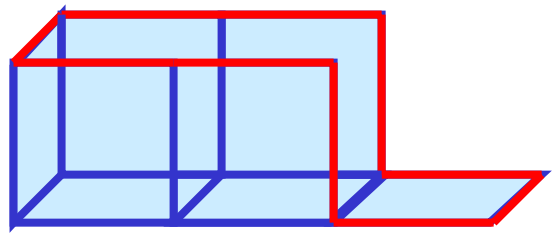




-- Logical operations: homologically nontrivial 2-surfaces of lattice and dual lattice:

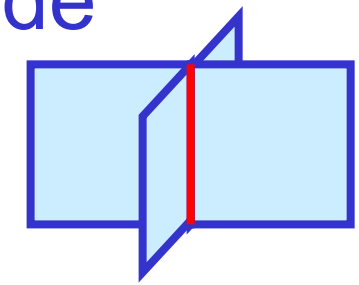


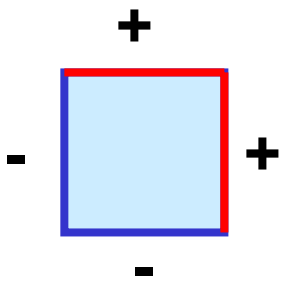
-- Defects are closed *loops of string,* (or dual loops) which bound droplets of flipped qubits:



Toom's rule for four-dimensional toric code

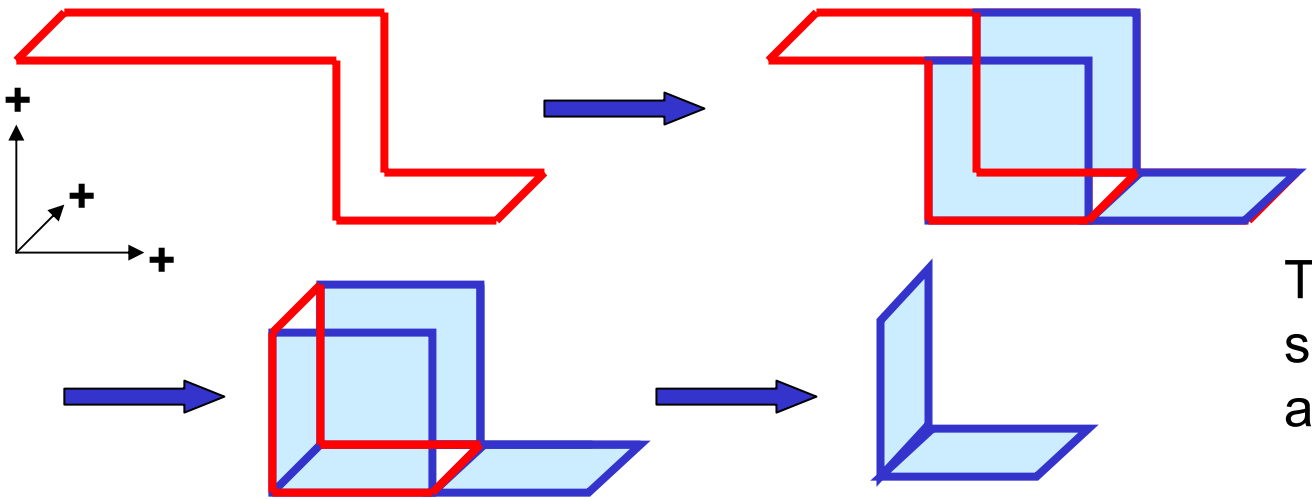
As in the 2D toric code, X error correction and Z error correction can be treated separately (lattice vs. dual lattice).





We can formulate a "Toom's rule" by defining + ("north" and "east") directions and – ("south" and "west") directions in each plane. In each recovery step, a cell flips if and only if string occupies both of its + edges. (Refreshable ancillas, used to measure syndrome, extract entropy introduced by noise.)

Under this rule (applied without noise), error "droplets" erode, and a string loop is removed in a time proportional to the loop's linear span. Example:

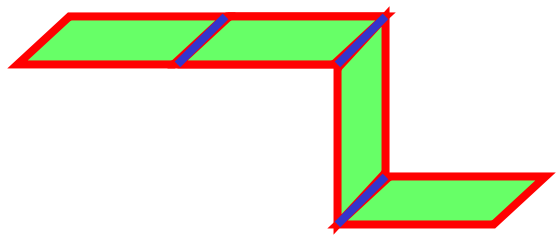


Thus, as in 2D, we can show that large strings are exponentially rare.

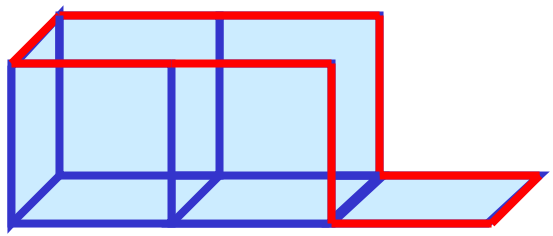
Toom's rule for four-dimensional toric code

In 4D, though Toom's rule removes strings, the cells afflicted by the noise might not be corrected. What can happen instead is that the flipped plaquettes form a (homologically trivial) closed surface. The operator that flips these plaquettes is in the code stabilizer, so the error recovery is successful.

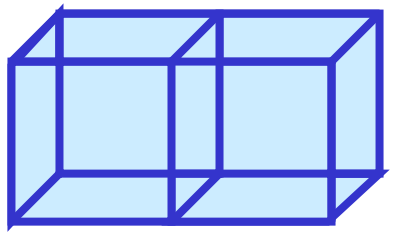
For example, if faults occur on these green plaquettes:



Toom's rule flips these blue plaquettes:

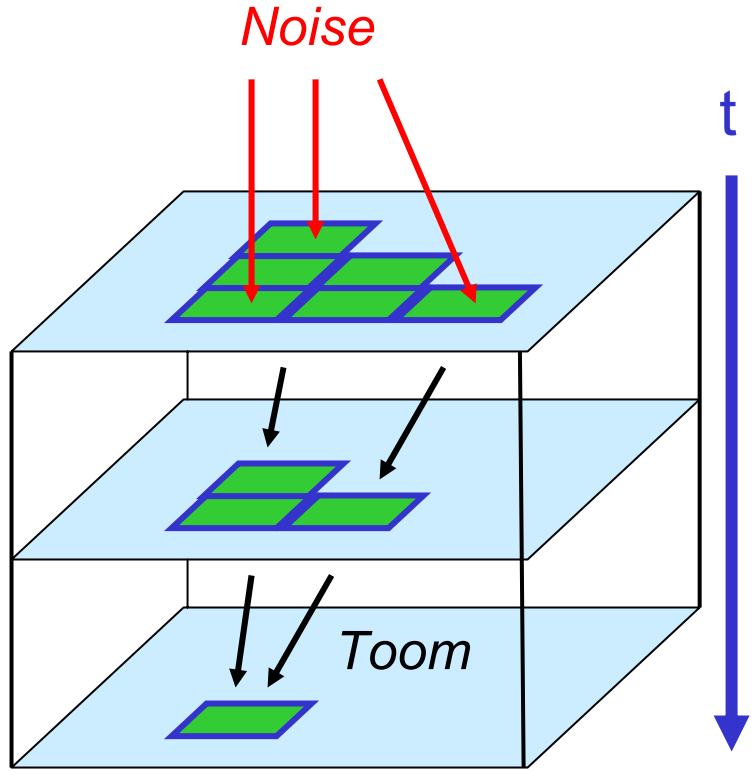


And the net result is that plaquettes are flipped on this closed surface:



Toom's rule for four-dimensional toric code

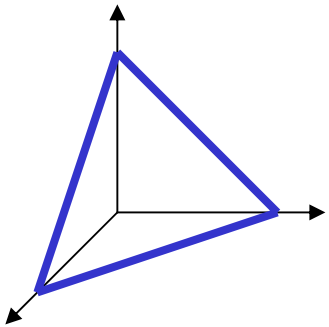
As in 2D, we can analyze the abundance of strings of size *s* by considering explanations for the one-plaquette loop of string that remains after *s*-1 iterations of the noiseless Toom rule. At least *n* = *s* faults are required



$$\text{Prob[string of size } s] \leq \sum_{n=s}^{\infty} |E_n| \, \varepsilon^n = O\left(\varepsilon^s\right)$$

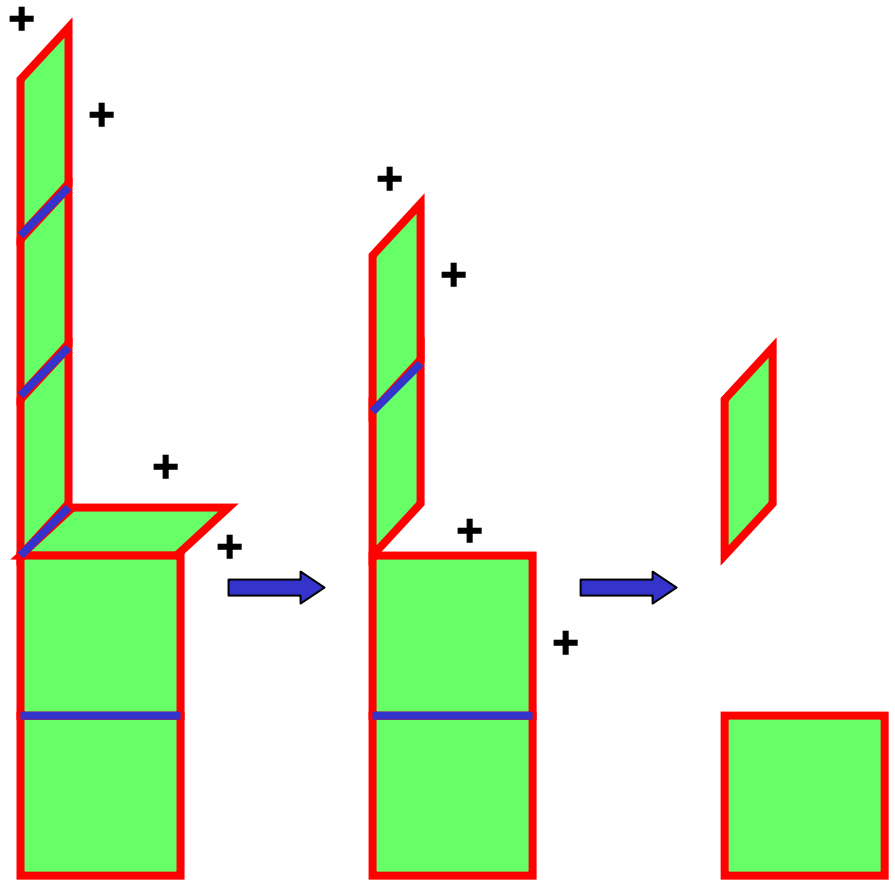
If the growth of $|E_n|$ is exponential, then for ε below threshold, large droplets are exponentially suppressed (as in the 2D PCA).

Toom's rule for four-dimensional toric code



A string has a "size," essentially the number of successive applications of Toom's rule that are needed to remove the droplet. This size is the side length of a triangular solid that contains the droplet. The droplet has *poles*, extremal cells in each of 5 directions.

As for the 2D PCA, a string in the four-dimensional toric code has a minimal explanation, characterized by events in which strings form due to faults, or in which the poles of pre-existing strings are joined by new faults. Again, each explanation corresponds to some tree, and these trees can be counted. Of course, because of the higher dimensionality, the trees are drawn on a graph of higher maximal degree than in the 2D case.



Another (not very important) new feature is that the Toom rule can cause a string loop to break into disconnected pieces:

Toom's rule for four-dimensional toric code

As for the 2D PCA, a string in the four-dimensional toric code has a minimal explanation, characterized by events in which strings form due to faults, or in which the poles of pre-existing strings are joined by new faults.

Explanations in which the cells damaged by noise are corrected by Toom's rule are counted by the same rules as for counting Toom graphs in 2D (but with a larger maximal degree appropriate to 4D). The main new feature in 4D is to account for the closed error surfaces that might be generated.

Thus, as we follow the (noisy) evolution of a string backward in time, we need to keep track of both the different ways in which the string configuration can arise as the boundary of a droplet of cells in error, *and* the ways in which these cells in error can be explained by pairs of cells in error on the previous time slice.

This only increases the number of ways that the pole of a string can evolve by a combinatoric factor. Therefore (as in 2D), the number of explanations with *n* faults grows in 4D no faster than exponentially with n:

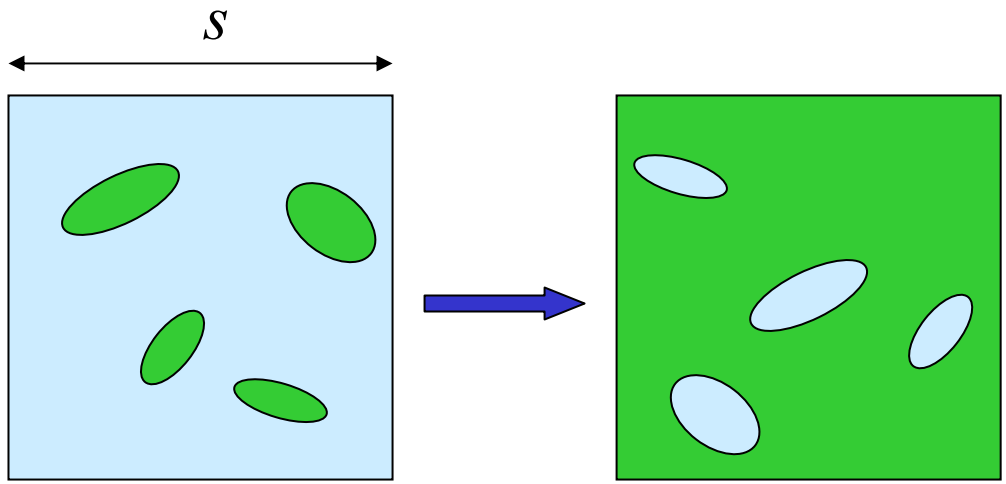
$$\text{Prob[string of size } s] \leq \sum_{n=s}^{\infty} |E_n| \, \varepsilon^n = O\left(\varepsilon^s\right)$$

Toom's rule for four-dimensional toric code

What is the probability of a encoded error?

We could decode by measuring all qubits, applying Toom's rule until all strings are removed, and then calculating the parity for some homologically nontrivial surface.

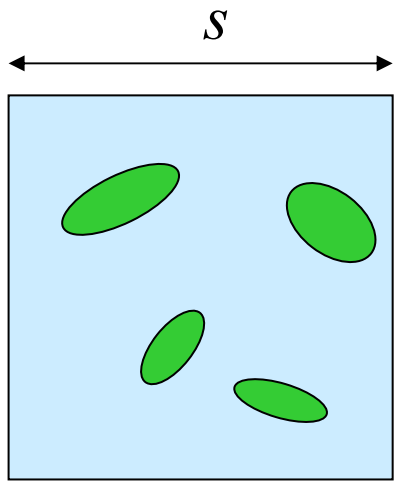
An encoded error might occur if a string loop gradually grows, eventually reaching a size comparable to the linear size of the code block. That's not likely, since large loops are suppressed.



Alternatively, many plaquettes must flip on some homologically nontrivial surface, all in one time step. That, too, is unlikely, since the number of surfaces grows only exponentially with the surface *area*. For a torus with linear size s, the probability of an encoded error per time step is:

$$\text{Prob[encoded error]} = O\left(\varepsilon^{s/16}\right)$$

Simulating a Clifford group circuit



For this (CSS) code, Clifford group gates (generated by CNOT, Hadamard, and phase gates) are transversal. A gate is simulated fault-tolerantly as the transversal gate followed by a recovery step.

Now we need to worry about propagation of error from one block to another during CNOT gates, but as for the simulation of a 1D PCA, the analysis of encoded errors needs little revision: we just increase the maximum degree of the graph on which the explanation tree is drawn, paying a modest price in the value of the threshold. (Error propagation during syndrome measurement can be controlled via fault-tolerant ancilla preparation.)

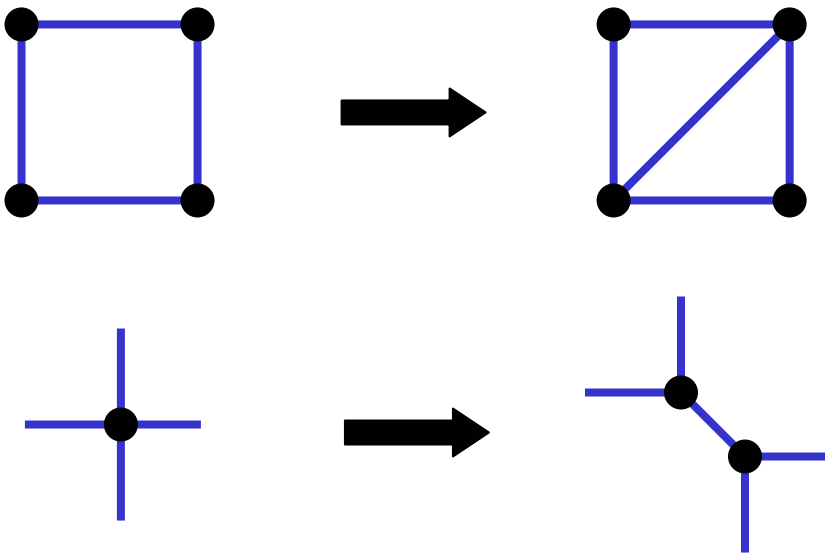
The depth blow-up is a constant. The error probability per gate is $e^{-O(s)}$; therefore, we can achieve an error probability per gate of (δ / L) by choosing linear size of the torus to be $s=O[\log(L/\delta)]$. The block size is $O(s^4)$, so

$$L^* = O(L \log^4 L) \quad D^* = O(D)$$

Completing the universal gate set

To complete the universal gate set, we need a fault-tolerant simulation of a quantum gate beyond the Clifford group. Such gates actually dominate the size and depth blow-ups of our circuit simulation.

We use a Dennis-Bravyi-Kitaev purification circuit to prepare the needed *quantum software*. The code block is built in stages, using CNOT gates that double the block size (by adding plaquettes that split an edge or divide a cube), followed by iterations of Toom's rule and a purification protocol that squares the error probability.



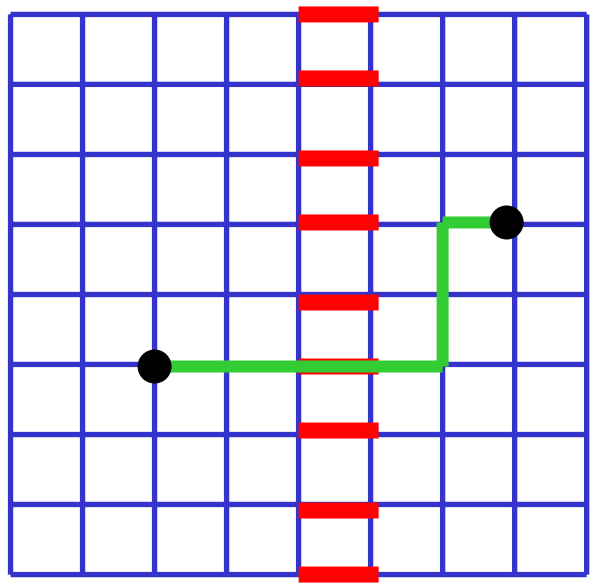
But … in the purification circuit and in the "teleportation" circuit that consumes the software, we must simulate an encoded measurement, followed by a Clifford group operation conditioned on the measurement outcome. Since the software preparation is done "off-line" it doesn't affect the depth. But the "measurement" that consumes the software dominates the blow-up:

$$L^* = O(L \log^5 L) \quad D^* = O(D \log\log L)$$

Fault-tolerant measurement

We need to measure the parity of the s^2 bits on a homologically nontrivial surface. This can be done by an ideal circuit with size $O(s^2)$ and depth $O(\log s)$. It can also be done by a fault-tolerant classical circuit in depth $O(\log s)$, *if* the bits are protected by a repetition code that has length $O(s)$.

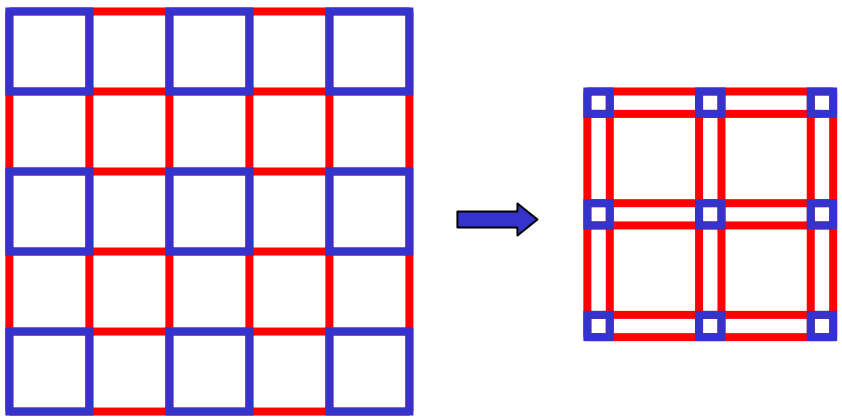
After "measuring" the bits, we can encode the bits fault tolerantly (using a repetition code) in depth O(log s). However …the parity could flip if the surface is crossed by an error droplet (i.e., if a string winds around the surface). Thus, we need to remove all the strings before the parity is measured.

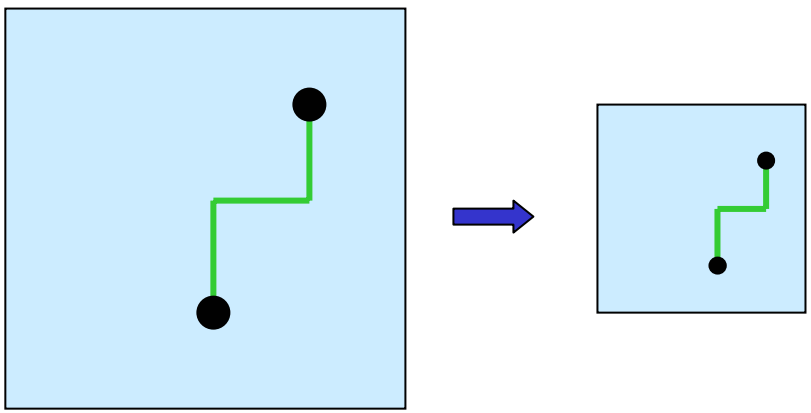


We could remove the strings by "measuring" all bits, encoding the outcomes, and then applying the (noiseless)Toom rule until strings of size $O(s)$ are gone. But that would take too long, requiring depth $O(s)$!

Fast fault-tolerant measurement

We need a faster decoding procedure, whose depth is logarithmic in the block size. This is possible with concatenated codes, because of their hierarchical organization. It can also be achieved with toric codes, using a *coarse-graining* procedure in which the toric block *contracts* while error protection is maintained.





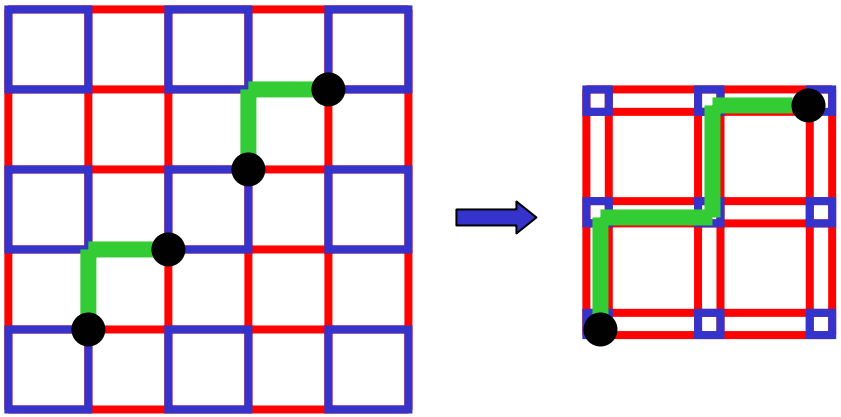
E.g., in 2D omit half the bits (the blue ones), by contracting plaquettes to sites, and compressing a pair of bits (red ones) to the parity of the pair. In 4D, contract a cube to an edge.

When the block contracts, large error droplets typically contract by about the same factor.

However, contraction of the block can also cause previously disconnected droplets to fuse together….

Fast fault-tolerant measurement

Contraction of the block can cause previously disconnected droplets to fuse together, so that the "comoving" size of the droplet grows:



This joining of droplets can be controlled (in 4D) by including a constant number of Toom iterations in between successive coarse-graining steps.

Thus we have a hierarchical decoding procedure (analogous to the decoding of a concatenated code). The Toom step shrinks all droplets and removes the smallest ones. In the coarse graining step the droplets contract while maintaining, roughly, the same comoving size, then the smallest of the remaining droplets are removed, etc.

When a 4D toric block with linear size s is decoded by this procedure, the probability that a string reaches, say, 1/16 the linear size of the contracting block remains exponentially small in s.

Thus we can decode with sufficient accuracy in depth $O(\log s)= O(\log \log L)$.

The cost of quantum fault tolerance

The outcome of the parity measurement is encoded using a length O(s) repetition code. In depth O(log s), the encoding can be expanded fault tolerantly to a length O(s^4) repetition code. Then a Clifford gate conditioned on the outcome can be executed transversally. This completes the simulation of our non-Clifford gate, and of our fault-tolerant repertoire.

Conclusion: Using a four-dimensional topological code, a local recovery procedure, a (nonlocal) hierarchical decoding procedure, and a (nonlocal) software purification protocol, we can simulate an ideal quantum circuit with size L and depth D, using a circuit of noisy (but not *too* noisy) quantum gates, with size and depth (Ahn & JP):

$$L^* = O(L \log^5 L) \quad D^* = O(D \log\log L)$$

Our arguments, described so far for 4D, can be generalized to any even dimension ≥ 4…

The cost of quantum fault tolerance

Our arguments, described so far for 4D, can be generalized to any even dimension ≥ 4…In higher dimensions the threshold is worse, but the size blow-up scales more favorably.

For the *d*-dimensional toric code (*d* even) qubits live on (*d*/2) cells, and encoded operations live on (*d*/2)-surfaces. The syndrome detects the (*d*/2 -1)-dimensional boundary of an error droplet. Encoded errors occur at the rate: $\exp\left[-O\left(s^{d/2-1}\right)\right]$

Therefore, chose $s^{d/2-1} = O(\log L)$

and block size $O\left(s^d\right) = O\left(\log^\gamma L\right) \quad ; \quad \gamma = \dfrac{d}{d/2-1} = \dfrac{2}{1-2/d}$

Simulating encoded measurement (using a classical repetition code of length $O(s^{d/2-1})$) blows up the size by another factor of log *L*, and the depth by log log *L*, so:

$$L^* = O(L \log^{\gamma+1} L) \quad D^* = O(D \log\log L)$$

The cost of quantum fault tolerance

- For a topological quantum memory in four (or more) dimensions, a local Toom's rule ensures that memory errors occur at an exponentially small rate (e.g. $e^{-O(s)}$ for block size $O(s^4)$ in four dimensions).

- Combining Toom's rule with transversal gates, we can simulate Clifford group computation with a constant depth blowup.

- We can reliably decode a block (storing the result in a classical repetition code) in depth $O(\log s)$.

- We can simulate a gate that completes our universal repertoire in depth $O(\log s)$.

- We can simulate an ideal quantum circuit in depth $D^*=O(D \log\log L)$

- Open question: Can the depth blow-up be reduced further to a constant? To achieve a constant-depth simulation, we need to overcome a significant difficulty: simulated measurements seem to be needed to simulate fault-tolerant universal gates, and since the quantum information is nonlocally encoded, we don't know how to measure an encoded block in constant depth. If constant depth can't be achieved, we have identified an essential difference between classical and quantum fault tolerance.