


Single-shot quantum error correction in intertwined toric codes

Charles Stahl *Department of Physics, University of Colorado, Boulder, Colorado 80309, USA*

(Received 7 May 2024; accepted 6 August 2024; published 23 August 2024)

We construct a subsystem code in three dimensions that exhibits single-shot error correction in a user-friendly and transparent way. As this code is a subsystem version of coupled toric codes, we call it the intertwined toric code (ITC). Although previous codes share the property of single-shot error correction, the ITC is distinguished by its physically motivated origin, geometrically straightforward logical operators and errors, and a simple phase diagram. The code arises from three-dimensional (3D) stabilizer toric codes in a way that emphasizes the physical origin of the single-shot property. In particular, starting with two copies of the 3D toric code, we add check operators that provide for the confinement of pointlike excitations without condensing the loop excitations. Geometrically, the bare and dressed logical operators in the ITC derive from logical operators in the underlying toric codes, creating a clear relationship between errors and measurement outcomes. The syndromes of the ITC resemble the syndromes of the single-shot code by Kubica and Vasmer, allowing us to use their decoding schemes. We also extract the phase diagram corresponding to ITC and show that it contains the phases found in the Kubica-Vasmer code. Finally, we suggest various connections to Walker-Wang models and measurement-based quantum computation.

DOI: [10.1103/PhysRevB.110.075143](https://doi.org/10.1103/PhysRevB.110.075143)

I. INTRODUCTION

Single-shot quantum error-correcting codes provide for a streamlined error-correction protocol, in which a single round of imperfect measurements suffices to correct errors [1]. The two-dimensional (2D) toric code [2] does not realize this property because the pointlike excitations cannot be reliably located in a single round of faulty measurements [3]. On the other hand, the excitations in the four-dimensional (4D) toric code are all extended excitations, allowing for single-shot error correction [3]. The toric codes are all stabilizer codes, meaning that all operators in the code commute. In stabilizer codes, single-shot error correction is closely related to self-correction, suggesting that any single-shot stabilizer code is, in fact, self-correcting. No single-shot stabilizer codes have been found below four dimensions.

On the other hand, single-shot *subsystem* codes do exist in three dimensions. Subsystem codes generalize stabilizer codes by having a check group, which does not commute, and a stabilizer group, which does commute. The gauge color code (GCC) [4] was the first three-dimensional (3D) subsystem code shown to be single-shot [1]. Despite the code's remarkable properties, its complicated construction meant that no other examples were found until seven years later, by Kubica and Vasmer [5]. Even in the Kubica-Vasmer code (KVC), the essential physical properties that provide for the success of single-shot error correction remain elusive.

Here we construct a single-shot 3D quantum error-correcting code that we call the intertwined toric code (ITC). This new code displays three main advantages: a transparent physical motivation and interpretation, geometrically simple logical operators, and a straightforward derivation of its phase diagram. Physically, the ITC draws inspiration from the 3D toric code. Its check group consists of the toric code operators

supplemented by single-site check operators. As a result, the ITC stabilizers detect the pointlike excitations and the check operators detect the stringlike operators that make those excitations, as depicted in Fig. 1(a). This clarifies that, physically, the ITC confines the pointlike excitations without condensing the extended excitations—which is not possible in stabilizer codes.

Measuring a different set of generators in the check group leads to check syndromes reminiscent of the KVC. In particular, the effective pictures of the syndromes in this second basis of the ITC contain two flavors of violated stabilizers connected by four flavors of check operators (whereas the GCC has eight stabilizers and 24 check operators). We illustrate one sector of this presentation of the ITC in Fig. 1(b). In this setting, the advantage of the ITC is the geometric simplicity of its logical operators: Membrane operators are defined on membranes and dual membranes on the lattice, and stringlike operators are defined on paths and dual paths on the lattice. Furthermore, it is easy to find a partial logical operator that creates any given measurement outcome, as described at the end of Sec. III A.

Readers with a background in condensed matter may view subsystem codes as corresponding to families of Hamiltonians, with multiple gapped phases of matter separated by phase transitions. Such analyses exist for the GCC [6] and the KVC [7]. The ITC phase diagram is particularly easy to extract, and we find all of the phases of the KVC along with one additional phase.

As an aside, the subsystem code literature uses the term “gauge operator” and “gauge group” instead of “check operator” and “check group.” To avoid confusion with the condensed-matter literature, where error-correcting codes often have interpretation in terms of gauge theories, we avoid using the word “gauge operator” herein. Instead, we refer to

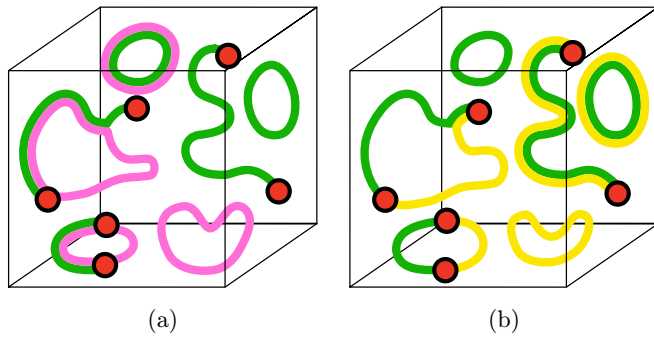


FIG. 1. Characteristic measurement outcomes in the ITC. (a) In the underlying toric code, green lines are truncated stringlike logical operators and pink lines are extended excitations. In the ITC, these objects are put on equal footing as lines of violated check operators (7). In both the toric code and the ITC, the red dots are violated stabilizers. (b) With the alternative basis of check operators (20), measurement outcomes consist of two colors of extended objects (green and yellow lines). These cannot end except on violated stabilizers, which must be the simultaneous endpoints of one line of each color.

the relevant operators as “check operators” in the subsystem code, because one “checks” the value of these operators during the error-correction procedure [8].

The remainder of this paper is organized as follows. In Sec. II we intertwine two toric codes, find a logical qubit, and describe the logical operators. We use a different set of check operators within the ITC in Sec. III to explore the possible measurement outcomes and outline the single-shot error correction procedure. In Sec. IV, we describe the phase diagram and explore connections of the various phases to previous literature. We conclude in Sec. V with some exploratory connections to other constructions and with a discussion of some open questions.

II. INTERTWINING TORIC CODES

To provide readers with some background, we start with a brief review of subsystem codes in Sec. II A. Then, to define the ITC, we define the check group and stabilizer group on a cubic lattice with periodic boundary conditions in Sec. II B. Like the GCC and the KVC, the ITC does not encode any logical qubits on closed manifolds. To encode logical qubits we define boundary conditions in Sec. II C. The boundary conditions furnish the ITC with logical qubits, and we describe the logical operators on those qubits in Sec. II D.

A. Brief review of subsystem codes

Recall that a stabilizer code consists of a stabilizer group \mathcal{S} , which is a commuting subgroup of the Pauli group that does not contain the global phase operator -1 . Elements of \mathcal{S} are called stabilizers. A state is a code state if it satisfies all of the stabilizers, which means that it is a $+1$ eigenstate of those operators. Logical operators are elements of the centralizer of \mathcal{S} , $\mathcal{Z}(\mathcal{S})$, which is the group of all Pauli operators that commute with \mathcal{S} . These operators map code states to code states. Of course, every stabilizer is in $\mathcal{Z}(\mathcal{S})$, but these are trivial logical operators. The nontrivial logical operators are

elements of $\mathcal{Z}(\mathcal{S})/\mathcal{S}$. Qubit errors are operators that do not commute with \mathcal{S} , and therefore take a code state out of the code space. A state not in the code space violates some stabilizers, meaning it is a -1 eigenstate of those stabilizers. A syndrome is the set of violated stabilizers in a given state, and an error causes a syndrome when it anticommutes with that set of stabilizers.

Subsystem codes [9] provide more flexibility than stabilizer codes by including a check group \mathcal{G} in addition to a stabilizer group. The check group does not need to commute and may contain global phase operators. Heuristically, we think of the stabilizer group as the center of \mathcal{G} , which is $\mathcal{Z}(\mathcal{G}) \cap \mathcal{G}$, or the group made up of operators in \mathcal{G} that commute with all operators in \mathcal{G} . However, this may contain pure phases, so the stabilizer group of a subsystem code is actually

$$\mathcal{S} = (\mathcal{Z}(\mathcal{G}) \cap \mathcal{G})/i, \quad (1)$$

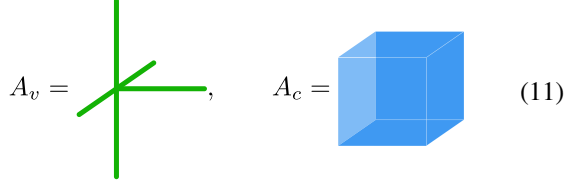
the center of \mathcal{G} with pure phases removed. Since $\mathcal{S} \subset \mathcal{G}$, each stabilizer is a product of check operators, allowing one to measure check operators in order to infer the value of stabilizers.

Code states cannot satisfy all of the check operators because \mathcal{G} does not commute. Instead, like in a stabilizer code, code states satisfy all of the stabilizers. Within the code space there are logical degrees of freedom (“logical qubits”) and other degrees of freedom (“check qubits”). The check operators act on the check qubits. Logical operators still must commute with all stabilizers, but come in two flavors depending on the degrees of freedom on which they act. Bare logical operators act only on the logical qubits while dressed logical operators act on the check qubits as well. Bare logical operators must commute with all check operators, making them elements of $\mathcal{Z}(\mathcal{G})$. Dressed logical operators need only commute with the stabilizers, making them elements of $\mathcal{Z}(\mathcal{S})$. Automatically, $\mathcal{G} \subset \mathcal{Z}(\mathcal{S})$ and $\mathcal{S} \subset \mathcal{Z}(\mathcal{G})$, so that the nontrivial bare and dressed logical operators are elements of $\mathcal{Z}(\mathcal{G})/\mathcal{S}$ and $\mathcal{Z}(\mathcal{S})/\mathcal{G}$, respectively.

The error-correction procedure consists of measuring the check operators, so the measurement outcome is the set of violated check operators. From a measurement outcome one can infer the stabilizer syndrome (the set of violated stabilizers). Code states have trivial stabilizer syndromes but still must have some nontrivial measurement outcomes. An operator that anticommutes with some check operators changes the measurement outcome, while an operator that anticommutes with some stabilizers causes a syndrome. In this language, dressed logical operators create trivial stabilizer syndromes but change the measurement outcome, while bare logical operators create trivial stabilizer syndromes and do not change the measurement outcome.

Subsystem codes provide a number of advantages over stabilizer codes. As in this paper, subsystem codes may be single-shot in three dimensions, while no known examples of such stabilizer codes exist. Subsystem codes also allow for “gauge fixing” [9] a procedure we explore in Sec. IV. They may allow for measuring smaller operators; for example, the subsystem toric code in Ref. [10] has check operators that only act on three qubits. Some further information on subsystem codes may be found in Ref. [11], which studies anyon theories in 2D subsystem codes, and Ref. [4].

The trivial (left) boundary follows simply from truncating our lattice in such a way that the boundary edges host qubits and boundary faces do not. We keep all the single-body X_e and Z_f check operators, along with any B_e and B_f check operators that remain four-body. For a vertex v and a boundary cube c on the trivial boundary, this results in the stabilizers



$$A_v = \text{green cross}, \quad A_c = \text{blue cube} \quad (11)$$

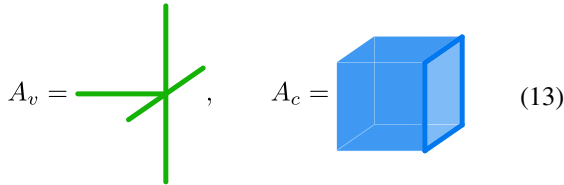
each of which acts on five qubits. Note that we can use these to construct membrane operators that span the periodic directions, so that there are no independent nonlocal stabilizers given these boundary conditions.

To construct the intertwined (right) boundary, truncate the lattice in the same way, with qubits on boundary edges but not faces. On every boundary edge we replace the single-site X_e check with the operators



$$K_e = X_e X_{f(e)} = \text{green square with edge}, \quad (12)$$

where $f(e)$ is the unique nonboundary face containing e in its boundary. The truncated A_v operators are still stabilizers, but the truncated A_c operators anticommute with some K_e operators. We can construct valid stabilizers by dressing each naive A_c with a B_f operator to create



$$A_v = \text{green cross}, \quad A_c = \text{blue cube} \quad (13)$$

for a vertex v and a cube c on the intertwined boundary.

The trivial and intertwined boundaries are related to each other by U_{CX} (9), in the sense that $K_e \leftrightarrow X_e$ under the symmetry. Thus, applying U_{CX} to the entire lattice simply swaps the two boundaries.

With these boundary conditions, the ITC encodes $K = 2$ logical qubits. The counting of both the check operators and the constraints is easiest on a lattice with length L in the periodic directions and length $L + 1$ in the other direction. There are $3L^3 + 2L^2$ qubits on edges and $3L^3 - L^2$ qubits on faces, for a total of $N = 6L^3 + L^2$ physical qubits. There are L^2 of the K_e operators on the intertwined boundary, leaving $3L^3 + L^2$ edges to host X_e operators, while all $3L^3 - L^2$ faces have Z_f operators. We find $L^3 + L^2 - 1$ independent A_v operators and $L^3 - 1$ independent A_c operators. There are $3L^3 + L^2$ total B_f operators and $L^3 + 1$ relations between them, giving $2L^3 + L^2 - 1$ independent B_f operators. There are $3L^3 - 2L^2$ total B_e operators and $L^3 - L^2 + 1$ relations between them, leaving $2L^3 - L^2 - 1$ independent B_e operators. Thus, there are

$$K = N - \frac{1}{2}(\log_2 |\mathcal{G}| + \log_2 |\mathcal{S}|) = 2 \quad (14)$$

logical qubits. In the next subsection we construct bare and dressed logical operators for one of the logical qubits.

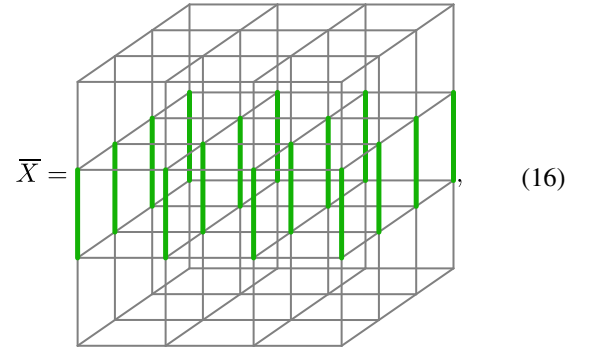
D. Logical operators

Recall that bare logical operators commute with all check operators (and therefore all stabilizers), while dressed logical operators commute with all stabilizers but not all check operators. In the bulk, we can construct membrane operators,

$$\bar{Z} = \prod_{f \in \mathcal{M}} Z_f, \quad \bar{X} = \prod_{e \in \mathcal{M}^*} X_e, \quad (15)$$

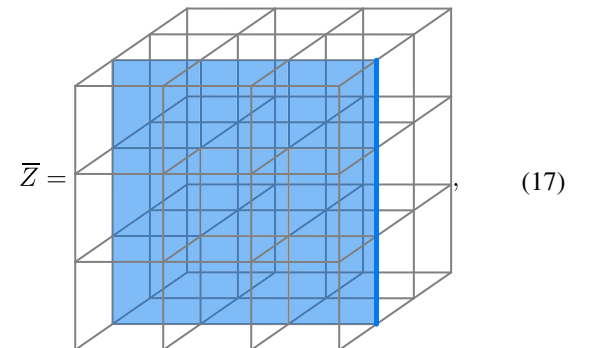
where \mathcal{M} and \mathcal{M}^* are a membrane and *dual* membrane, respectively. If these membranes are contractible, then the associated operators are products of local stabilizers and therefore trivial logical operators. If the membranes are closed but noncontractible, then the operators are (possibly nonlocal) stabilizers, as previously discussed. To define logical operators, the membranes must terminate on open boundaries.

At the trivial boundary, all bare logical operators may terminate without violating any check operators. At the intertwined boundary, the naive termination of \bar{X} commutes with all check operators, resulting in



$$\bar{X} = \text{green lines in grid}, \quad (16)$$

where, recall, the lattice is periodic in the top/bottom direction and the front/back direction. This operator is *not* a product of check operators (and therefore not a stabilizer) because the single-site X_e operators on the intertwined boundary is not in the check group \mathcal{G}_{ITC} (12). However, the naive termination of \bar{Z} anticommutes with the check operators K_e for every edge e in $\mathcal{C} = \partial \mathcal{M} \cap \text{Int}$, the intersection of the membrane boundary with the intertwined boundary. We can fix this by dressing the \bar{Z} membrane operator with $\prod_{e \in \mathcal{C}} Z_e$, as in



$$\bar{Z} = \text{blue shaded region in grid}, \quad (17)$$

which satisfies all the K_e check operators. It is straightforward to verify that \bar{X} and \bar{Z} anticommute, showing that they are the bare logical operators for one of the two logical qubits.

The logical operators for the other logical qubit locally look like \bar{X} and \bar{Z} , but are stretched in the opposite directions. If we name the new logical operators \bar{X}_2 and \bar{Z}_2 , then \bar{X}_2 is now periodic in the top/bottom direction and \bar{Z}_2 is periodic in the front/back direction. This confirms that the ITC with periodic boundary conditions in the top/bottom direction and the front/back direction encodes two logical qubits. In the Appendix we define the e -condensed (m -condensed) boundary, where only the logical Z operator (logical X operator) can terminate. We use these to define the ITC with only a single logical qubit.

The dressed logical operators are stringlike operators $\tilde{Z} = \prod_{e \in \mathcal{C}} Z_e$ and $\tilde{X} = \prod_{f \in \mathcal{C}^*} X_f$, where \mathcal{C} and \mathcal{C}^* are a noncontractible curve in the top/bottom direction and a noncontractible *dual* curve in the front/back direction, respectively. These operators anticommute with the bare logical operators \bar{X} and \bar{Z} , but commute with each other. This is possible because they act on the logical qubits *and* the nonlocal “check qubits,” as described previously.

In fact, the dressed logical operators are so named because they can be written as the bare logical operators dressed with check operators. For example, the Z_f operators in (17) are all check operators. Multiplying \tilde{Z} by the individual Z_f operators removes them, leaving behind just the line of Z_e operators on the intertwined boundary. This is a dressed logical operator \tilde{Z} . We can obtain other dressed logical operators by multiplying \tilde{Z} by various B_f check operators, moving it into the bulk.

Whereas one of the logical operators in the 3D stabilizer toric code is inherently membrandlike, the ITC possesses stringlike (dressed) logical operators for both sectors. Thus, we have traded some stability in one sector for extra stability in the other. In the next section, we will show how this stability results in detectability of partial logical operators.

III. ERRORS AND ERROR CORRECTION

Having defined the check operators, stabilizers, and logical operators, we are prepared to describe the results of both qubit errors and measurement errors. We focus on Z -type qubit errors and X -type measurement errors, as the description of the other types follows analogously. One option is to describe the measurement outcomes of the B operators and single-site check operators defined previously. This would result in measurement outcomes like those in Fig. 1(a), with green for violated X_e and pink for violated B_e . Instead, here we use a basis of \mathcal{G}_{ITC} that more closely mirrors the KVC, allowing us to use their decoder. We are only going to choose a different set of generators for the check group, leaving the check group itself unchanged, along with the lattice, geometric boundary conditions, stabilizers, and logical operators. Thus, we really are analyzing the same ITC, just with a different presentation.

On the same lattice, with qubits on edges and faces, define the operators

$$K_e = X_e B_e = X_e \prod_{f \in \partial^* e} X_f, \quad (18)$$

$$K_f = Z_f B_f = Z_f \prod_{e \in \partial f} Z_e, \quad (19)$$

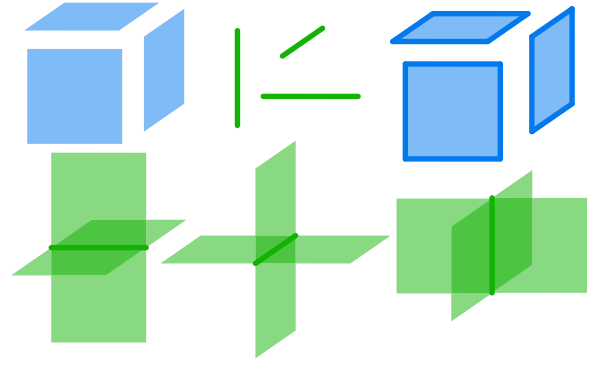


FIG. 2. Check operators in an alternative presentation of the ITC. Green and blue represent X and Z operators, respectively. Solid bars are edges, and shaded parallelograms are faces. On the top row we have the operators Z_f , X_e , and K_f , while on the bottom row we have the K_e check operators.

shown in Fig. 2. The check group

$$\mathcal{G}_{\text{ITC}} = \langle X_e, Z_f, K_e, K_f \rangle \quad (20)$$

is the same as \mathcal{G}_{ITC} in (7). The stabilizers are still the vertex and cube stabilizer operators,

$$A_v = \prod_{e \in \partial^* v} X_e = \prod_{e \in \partial^* v} K_e, \quad (21)$$

$$A_c = \prod_{f \in \partial c} Z_f = \prod_{f \in \partial c} K_f, \quad (22)$$

from (8), as they had to be. The relations (21) and (22) between check operators provides redundancy that we lacked in the previous section. They allows us to detect measurement errors in addition to qubit errors, which improves the decoding procedure.

On the trivial boundary we have the same check operators as before. For a boundary face, the K_f operator truncates to a four-body operator equivalent to B_f . For a boundary edge, naive truncation of K_e leads to two-body operators $X_e X_{f(e)}$ which we choose to discard. We instead keep X_e for each boundary edge, resulting in the same set of check operators as before and the same stabilizers as in (11).

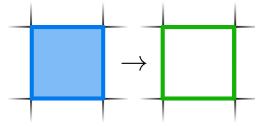
On the intertwined boundary, we make the other choice and keep the two-body operator $K_e = X_e X_{f(e)}$, justifying the name we previously gave this operator. Thus, we arrive at the same check operators and stabilizers as previously in (12) and (13).

In this section we describe the measurement outcomes of the KVC-inspired basis of measurements for the ITC. In Sec. III A we explore the ideal measurement outcomes that result from applying local qubit errors, using linear maps defined in Ref. [5]. Then we apply some logical operators transversally (as a series of local errors) in Sec. III B and see which check operators they violate along the way. In Sec. III C we outline how to use the measurement outcomes to perform single-shot error correction, even in the presence of measurement errors.

A. Local errors

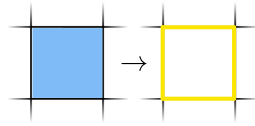
Recall that any code state satisfies all stabilizers but violates many check operators. To construct a generic code state, start in a state $|X_0\rangle$ that satisfies all stabilizers and all X -type check operators. This state exists, as all these operators commute. Then, create a generic code state by acting with Z -type check operators on $|X_0\rangle$. The resulting state is still a code state and satisfies all stabilizers, but violates some X -type check operators. More specifically, consider a Z -type check operator η and the linear map δ_M , which maps η to the set of check operators that anticommute with it. The nontrivial measurement outcome of X -type measurements on the state $\eta|X_0\rangle$ is $\delta_M\eta$.

To illustrate this, let the Z -type check operator be $\eta = K_f$ so that the measurement outcome is $\delta_M\eta = \prod_{e \in \partial f} X_e$, four X_e operators around the face f . Graphically, this relationship is



$$(23)$$

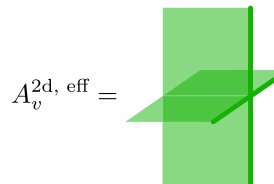
with η on the left and $\delta_M\eta$ on the right. Similarly, the Z_f check operator anticommutes with the four K_e operators on edges around the face f . This relationship is



$$(24)$$

where yellow lines are violated K_e check operators. By stacking many check operators we can create the measurement outcome of a generic code state, which consists of many green and yellow lines, all of which must form closed loops.

For a vertex v on the trivial boundary (chosen to be on the left-hand side) we have to measure the additional operator



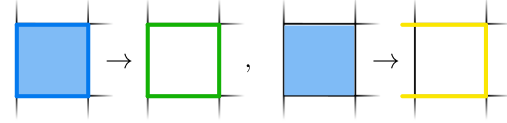
$$(25)$$

which can be shown to be in the check group. Measuring this operator allows for two redundant ways to construct the A_v operator out of measurement outcomes for v on the trivial boundary.¹ In particular, the redundancy is

$$A_v = \prod_{e \in \partial^\dagger v} X_e = A_v^{2D, \text{eff}} K_{e(v)}, \quad (26)$$

where $e(v)$ is the unique nonboundary edge in $\partial^\dagger v$. Let us draw violated $A_v^{2D, \text{eff}}$ operators as short yellow lines sticking out of

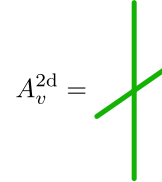
the lattice. Green lines still cannot end, but we can see from



$$(27)$$

that yellow lines can end if they extend to the left of the boundary.

On the intertwined boundary (on the right side) we must measure the redundant operator

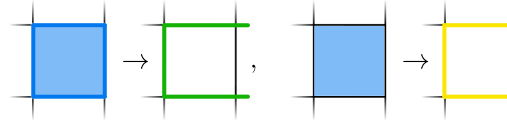


$$(28)$$

analogous to (25). The redundancy in stabilizer operators is

$$A_v = \prod_{e \in \partial^\dagger v} K_e = A_v^{2D} X_{e(v)}, \quad (29)$$

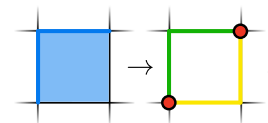
for a vertex v on the intertwined boundary. Let us draw violations of A_v^{2d} as green lines extending beyond the boundary. Measurement outcomes such as



$$(30)$$

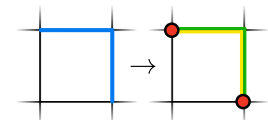
tell us that now green lines can end if they extend to the right of the boundary. To summarize, the violated check operators in any code state consist of closed green (X_e) and yellow (K_e) loops in the bulk. Yellow loops may end at the trivial boundary and green lines may end at the intertwined boundary.

To leave the code space, act with an operator that anticommutes with some stabilizers. Any such operator, called an error, is not in the check group. We can extend the map δ_M to map an error ϵ to the check operators $\delta_M\epsilon$ that it anticommutes with. Another map, called ∂_S , maps an error to a stabilizer syndrome, the set of stabilizers it anticommutes with. We can represent both maps graphically, simultaneously. For example, in



$$(31)$$

we have ϵ on the left and both $\delta_M\epsilon$ and $\partial_S\epsilon$ on the right. The Z -type error on the left anticommutes with two X_e check operators (green), two K_e check operators (yellow), and two A_v stabilizers (red). A different error,



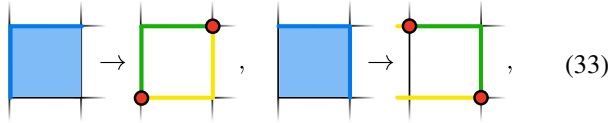
$$(32)$$

also maps to two X_e check operators, two K_e check operators, and two A_v stabilizers.

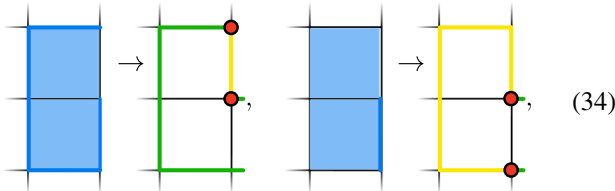
¹Thank you to Yaodong Li for pointing out the necessity of measuring this operator.

Composing similar errors creates generic syndromes and measurement outcomes, subject to some constraints called relations. The relations say that green lines and yellow lines can end only on red points, and red points must have an odd number of green lines and an odd number of yellow lines incident upon them. Phrased in terms of operators, the relation is that, for any vertex v , the number of violated K_e and X_e operators on the edges in $\partial^+ v$ must have the same parity. If that parity is odd, then the stabilizer A_v is violated, and if the parity is even, then A_v is satisfied. These relations follow from (21) and (22). In other words, violated stabilizers are connected by violated check operators, so that they are no longer purely pointlike errors. In Sec. III C we show how these connecting lines allow us to probabilistically determine the locations of the true qubit errors, even in the presence of measurement errors.

On the trivial boundary we know that yellow lines can end without violating stabilizers. Errors cause measurement outcomes that look like



which show that violated stabilizers still need green and yellow lines attached to them, but yellow lines may end without a corresponding violated stabilizer to the left of the boundary. The intertwined boundary instead allows configurations like



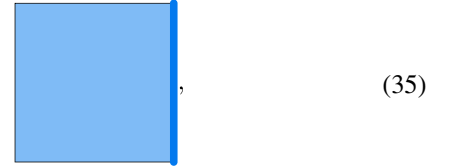
so that green lines can end without a corresponding violated stabilizer. To summarize, red dots (violated A_v syndromes) need to be connected to green and yellow lines. Yellow lines may end without a violated stabilizer at the trivial boundary, and green lines may end without a violated stabilizer at the intertwined boundary.

In the coarse-grained description of the ITC, any Z -type error consists of open strings of Z_e operators, possibly decorated by open or closed membranes of Z_f operators and closed strings of Z_e operators. A bare string of Z_e operators anticommutes with X_e and K_e check operators along its length. A bare membrane of Z_f operators anticommutes with K_e check operators around its boundary, and a Z_f membrane dressed with Z_e operators around its boundary anticommutes with X_e check operators around that boundary. Open Z_e strings violate A_v stabilizers at their endpoints. These rules allow us to invert the δ_M map and find a representative error that can cause any valid measurement outcome. For example, the measurement outcome represented graphically in Fig. 1(b) could be caused by the error shown in Fig. 4(a).

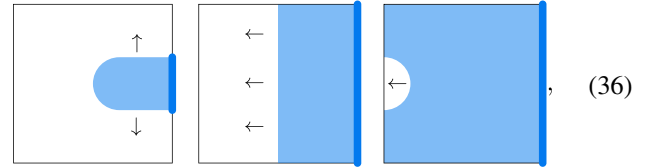
B. Transversal logical errors

Isolated errors are not so dangerous. Problems arise when errors conspire to mimic logical operators. For example, applying a string of Z_e errors is the same as applying a partial Z -type dressed logical operator. Applying a logical operator this way, as a series of local operators, is called a transversal decomposition of the logical operator. At any point in the decomposition, the partial logical operator results in violated A_v stabilizers at its endpoints. Thus, a transversal decomposition of the Z -type dressed logical operator transports violated A_v operators across the system, leaving behind violated X_e and K_e check operators. This subsection is a pedagogical description of the transversal decomposition of the bare logical Z operator.

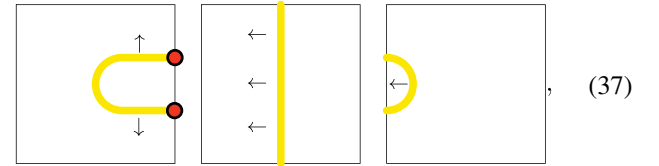
Recall the Z -type bare logical operator



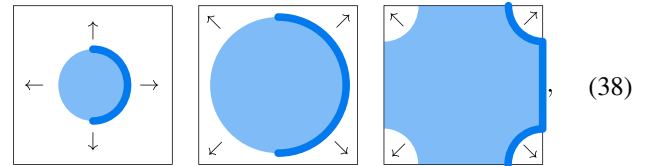
drawn as a slice through the entire lattice. The figure represents a coarse-grained version of the one in (17). We could apply this operator transversally in many different ways. We could truncate it from right to left,



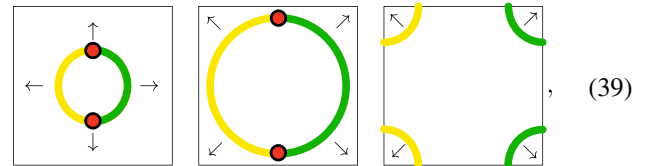
resulting in the measurement outcomes



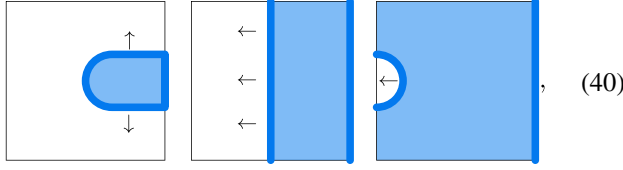
where the noncontractible strings of violated K_e check operators witnesses the partial logical operator. A geometrically different transversal decomposition of the bare logical operator,



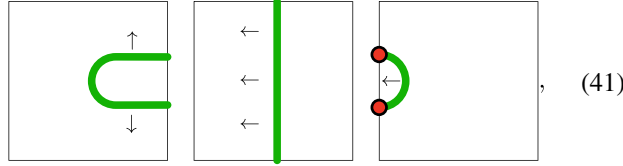
results in the measurement outcomes



where there are noncontractible strings of violated K_e and X_e check operators stretching from one boundary to the other. A different transversal decomposition of \tilde{Z} uses a different boundary,



resulting in the measurement outcomes



where the noncontractible strings now consist of violated X_e checks. Note that the violated stabilizers can appear on the left or right boundary or in the bulk, depending on the decomposition.

What is constant is that all decompositions involve transporting a pair of violated stabilizers vertically across the system and then removing the resulting violated check operators at the left and right boundary. Thus, either type of logical operator transports violated stabilizer across the system, and the difference is that dressed logical operators leave violated check operators behind while bare logical operators remove the violated check operators at the boundary.

C. Single-shot error correction

The point of any error-correction procedure is to prevent local errors from accumulating into logical errors. To that end, we want to detect and undo any partial logical operators. The simplest way is to find all violated stabilizers and pair them up to correct them while introducing the fewest residual errors. One way to do this in the stabilizer toric code is the minimal-weight perfect matching (MWPM) procedure [3].

The MWPM procedure successfully decodes the stabilizer toric code as long as measurements are ideal. However, incorrect measurements, where the wrong measurement outcome is written down, completely ruin the MWPM procedure for pointlike stabilizers. The problem is that a small number of incorrect measurements can lead to a large number of residual errors.

Connecting violated stabilizers with violated check operators, as in the GCC, the KVC, and the ITC, helps to recover from measurement errors. In fact, the syndromes of the ITC look like the syndromes of the KVC, at least when coarse-grained as in Fig. 3. This allows us to use the single-shot version of the MWPM procedure introduced in Ref. [5].

The single-shot MWPM procedure consists of two steps, which we describe first heuristically and then in more mathematical detail. The first step deals with violations of the relations from Sec. III A, which are unpaired endpoints of the measurement outcomes. These violations, collectively called the relation syndrome, are aphysical and can result only from measurement error. For example, if the true measurement

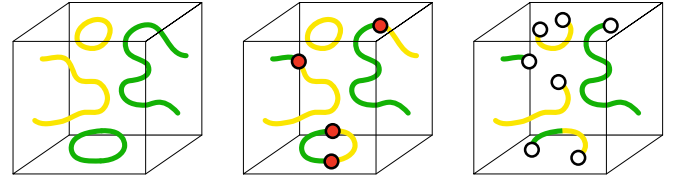


FIG. 3. Syndromes in the KVC-inspired presentation of the ITC. On the left we have a syndrome for a code states with no qubit or measurement errors. The middle figure displays qubit errors, which are violated stabilizers represented by red dots, but no measurement errors. On the right we have measurement errors, which are violated relations represented by empty circles.

outcome is a closed line of violated X_e check operators, some false negatives along this line result in endpoints of the line. The relation syndromes are the white circles in Fig. 3. The first round of MWPM pairs up these violated relations, inferring the shortest strings of measurement errors that could have caused them.

After inferring the measurement errors, we are left with valid measurement outcomes, with their attendant stabilizer syndromes. Recall that violated stabilizers are vertices where X_e and K_e lines simultaneously end. These are marked with red dots in Fig. 3. The second round of MWPM pairs up the violated stabilizers, matching the stabilizers in a way that is robust to measurement errors. Finally, error correction proceeds by acting with operators that remove the violated stabilizers.

To describe the process in more detail, the next part of this section unifies the entire error-correction procedure as a series of linear maps between vector spaces, closely following the discussion of single-shot error correction in Ref. [5]. We outline the important steps, highlighting the places where the ITC differs from the KVC.

We have been describing the ITC in terms of sets and groups: the set of qubits, the check group, the stabilizer group, the set of relations, etc. To formalize the decoding problem, we redefine these as vector spaces over \mathbb{F}_2 , the field with two elements. These spaces are the space of Z -type check operators C_G , the space of qubits C_Q , the space of X -type check operators C_M , the space of stabilizers C_S , and the space of relations C_R . For example, an element of C_S corresponds to an element of the stabilizer group, and an element of C_Q corresponds to a subset of the qubits. An “error” involves a Z -type qubit error $\epsilon \in C_Q$ and a measurement error $\mu \in C_M$, while a “syndrome” includes both the violated stabilizers $\sigma \in C_S$ and the violated relations in $\omega \in C_R$.

We already defined some linear maps between these spaces in Sec. III A: δ_M maps an error $\epsilon \in C_Q$ to the set of check operators $\delta_M \epsilon \in C_M$ that it anticommutes with, δ_S maps a measurement outcome $\zeta \in C_M$ to the inferred syndrome $\delta_S \zeta \in C_S$ and δ_S maps an error ϵ to the syndrome $\delta_S \epsilon \in C_S$ that it causes. These maps obey

$$\delta_S = \delta_S \delta_R \quad (42)$$

by definition. We also define some new maps: ∂_Q maps a Z -type check operator $\gamma \in C_G$ to the set of qubits $\partial_Q \gamma \in C_Q$ on which it acts, and δ_R maps a measurement outcome ζ to its relation syndrome $\delta_R \zeta \in C_R$.

The above-defined maps must obey some constraints coming from the definition of subsystem codes. First, the constraint

$$\partial_S \partial_Q = 0 \quad (43)$$

says that the Z -type check operators commute with the X -type stabilizers. Second, the constraint

$$\delta_R \delta_M = 0 \quad (44)$$

says that physical errors do not cause relation syndromes. These constraints are all encoded in a commutative diagram taken directly from Ref. [5],

$$\begin{array}{ccccc}
 & & \text{X-type check} & & \\
 & & \text{measurements} & & \\
 & & C_M & & \\
 & \nearrow (\delta_M | I) & & \searrow \begin{pmatrix} \delta_S \\ \delta_R \end{pmatrix} & \\
 C_G & \xrightarrow{\begin{pmatrix} \partial_Q \\ 0 \end{pmatrix}} & C_Q \oplus C_M & \xrightarrow{\begin{pmatrix} \partial_S & \delta_S \\ 0 & \delta_R \end{pmatrix}} & C_S \oplus C_R \\
 \text{Z-type} & & \text{qubits and X-type} & & \text{stabilizers} \\
 \text{check operators} & & \text{measurements} & & \text{and relations}
 \end{array} \quad (45)$$

where the block matrix structure of the maps follows from treating elements of the vector spaces as column vectors.

In the ITC, these spaces and maps decompose even further, simplifying the discussion. For example, the space of Z -type check operators is $C_G = C_{Z_f} \oplus C_{K_f}$, the direct sum of the spaces of Z_f and K_f check operators. The space of qubits is $C_Q = C_E \oplus C_F$, the direct sum of the spaces of edge qubits and face qubits, with C_{Z_f} , C_{K_f} , and C_F all isomorphic to the space of faces in the lattice. The space of X -type check operators decomposes into $C_M = C_{X_e} \oplus C_{K_e}$, the direct sum of the spaces of X_e and K_e check operators, with C_{X_e} , C_{K_e} , and C_E all isomorphic to the space of edges on the lattice. The space of stabilizers C_S and the space of relations C_R are both isomorphic to the space of vertices.

Each block of the block matrices in (45) also decomposes into a block structure. Furthermore, the individual components are simply the identity I and the geometric boundary operator ∂ in the bulk, although these may have to be truncated or supplemented at either boundary. The first column of

$$\partial_Q = \left(\begin{array}{c|c} 0 & \partial \\ \hline I & I \end{array} \right) \quad (46)$$

says that the Z_f check operator acts on a single face qubit, while the second column says that K_f acts on a single face qubit and the four edge qubits that surround it. Similarly, the first column of

$$\delta_M = \left(\begin{array}{c|c} I & 0 \\ \hline -I & \partial \end{array} \right) \quad (47)$$

says that a Z_e error anticommutes with the X_e and K_e check operator on that edge, while the second column says that a Z_f error anticommutes with the four K_e check operators around that face. Equations (23) and (24) represent (47) graphically. The two maps

$$\delta_S = (\partial \mid 0), \quad \delta_R = (\partial \mid \partial) \quad (48)$$

say that violated stabilizers are vertices with an odd number of violated X_e check operators around them, while violated relations are vertices with an odd total number of violated X_e and K_e check operators around them, respectively. The last map, $\partial_S = (\partial \mid 0)$, maps a set of qubit errors directly to the stabilizers it violates.

The decompositions automatically satisfy the constraints $\partial_S \delta_M$, $\partial_S \partial_Q = 0$, and $\delta_R \delta_M = 0$. The point of including this whole decomposition is to show that the decoding maps in the ITC are nicely geometric, like the decoding maps in the stabilizer toric codes.

The decoding strategy takes two steps. As described previously, code states have nontrivial syndromes generated by $\gamma \in C_G$, a set of Z -type check operators. The initial γ and some errors $\epsilon \oplus \mu \in C_Q \oplus C_M$ produce a measurement outcome

$$\zeta = \delta_M \epsilon + \mu + \delta_M \partial_Q \gamma \in C_M. \quad (49)$$

This measurement outcome violates a set of relations $\delta_R \zeta$, allowing us to infer a measurement error $\hat{\mu}$ with the same relation syndrome $\delta_R \hat{\mu} = \delta_R \zeta$. The resulting inferred stabilizer syndrome is $\delta_S(\zeta + \hat{\mu})$, allowing us to infer a qubit error $\hat{\epsilon}$ such that $\partial_S \hat{\epsilon} = \delta_S(\zeta + \hat{\mu})$. After correcting for the inferred error, the residual error is $\epsilon + \hat{\epsilon}$. If the decoding procedure succeeds perfectly, we have $\hat{\epsilon} = \epsilon + \partial_Q \gamma'$, where $\gamma' \in C_G$, so that the residual stabilizer syndrome is $\delta_S(\epsilon + \hat{\epsilon}) = \partial_S \partial_Q \gamma' = 0$. This is the doubled MWPM procedure from Ref. [5]. Even if the decoding procedure does not proceed perfectly, the residual stabilizer syndrome should be equivalent to one caused by a small number of qubit errors.

An advantage of the ITC is that both rounds of MWPM occur on the graph defined by the edges and vertices of the cubic lattice. In both cases, the violations appear on vertices and the inferred errors appear on edges. Similarly, for X -type qubit errors and Z -type measurement errors, the violations appear on cubes and the inferred errors appear on faces. This is geometrically dual to the previous cases. In all cases, the

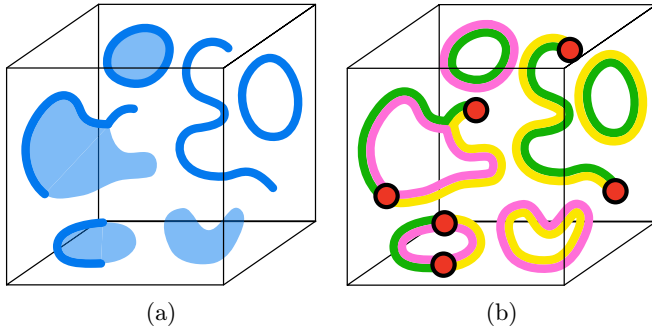


FIG. 4. An error in the ITC and its overcomplete syndrome. (a) If shaded blue areas are Z_f operators, and blue lines are Z_e operators, this is the error causing the syndromes shown in Fig. 1. (b) Measuring an overcomplete set of check operators leads to maximally redundant syndromes, where every line must be coincident with a line of another color. Note that removing yellow lines gives Fig. 1(a), while removing pink lines gives Fig. 1(b).

problems of finding $\hat{\mu}$ and $\hat{\epsilon}$ simply reduce to MWPM on the cubic lattice. At this point we can rely on the results from Ref. [5] to show that the ITC is indeed capable of single-shot error correction.

Moving beyond the analysis of Ref. [5], we can achieve maximal redundancy by measuring all six types of check operators: X_e, Z_f, K_e, K_f, B_e , and B_f . This set is overcomplete in the sense that to every edge there corresponds three check operators, with a nontrivial relation between them: $K_e X_e B_e = 1$. We can use the previous analysis to infer the relations for X -type check operators in the bulk. Every edge must be occupied by zero or two violated check operators. Lines of violated B_e check operators may not end, while K_e and X_e lines end only on violated stabilizers. Violated stabilizers must be simultaneous endpoints of K_e and X_e lines. These rules are illustrated in Fig. 4. On the trivial boundary, K_e and B_e lines may end without violated stabilizers, and violated stabilizers need only to be endpoints of X_e lines. On the intertwined boundary, X_e and B_e lines may end without violated stabilizers, and violated stabilizers need only to be endpoints of K_e lines.

In this overcomplete version of the ITC, the nontrivial relation $K_e X_e B_e = 1$ for every edge e means that a string of measurement errors is detectable everywhere along its length, rather than just at its endpoints. This could possibly lead to a more accurate while still efficient decoder. Of course, measuring overcomplete sets of check operators can always increase the accuracy of a decoder. This realization is not new, but might help in understanding the ITC.

IV. PHASE DIAGRAM OF THE ITC

So far, we have seen that the check group \mathcal{G}_{ITC} includes some terms that we might recognize from our favorite Hamiltonians. By construction, the B terms come from the 3D toric code Hamiltonian [12]. It is easy to see that the single-site terms form a paramagnet Hamiltonian. In addition, the K terms are equivalent to terms in the Raussendorf-Bravyi-Harrington (RBH) Hamiltonian [13] after a Hadamard

transformation on face qubits, mapping $X_f \leftrightarrow Z_f$. In this section we show that these three Hamiltonians live in the phase diagram corresponding to the ITC subsystem code.

A. Gapped phases in the ITC

Recall that, from a stabilizer code with stabilizer group \mathcal{S} , we can construct a gapped Hamiltonian,

$$H_{\mathcal{S}} = - \sum_{S \in \mathcal{S}} J_S S, \quad (50)$$

by including all the stabilizers with negative coefficients $-J_S < 0$. The coefficients for operators with large support should be small. Because the stabilizers all commute, the relative strengths of the coefficients do not matter. Thus, the stabilizer code corresponds to a single phase of matter. We can try to build a Hamiltonian from a subsystem code analogously, by including all the stabilizers and check operators with coefficients. The most general such Hamiltonian is

$$H_{\mathcal{G}} = - \sum_{S \in \mathcal{S}} J_S S - \sum_{G \in \mathcal{G}} J_G G + \text{H.c.}, \quad (51)$$

where we should again be careful to bound the strength of the coefficients for operators with large support [11]. The stabilizer coefficients J_S should be negative, but the check group contains pure phases, so there is no reason to impose negativity on the check coefficients J_G .

Since the check group is non-Abelian, the Hamiltonian $H_{\mathcal{G}}$ is frustrated and different choices of coefficients may give rise to different gapped phases. Thus, the subsystem code corresponds to a whole phase diagram with different phases and phase transitions. Phase diagrams have been studied for the GCC [6] and the KVC [7]. Additionally, subsystem codes can lead to novel constructions for 2D Abelian phases [11].

To simplify the analysis of the ITC Hamiltonian, define the Hamiltonian

$$\begin{aligned} H_{\text{ITC}} = & -J_{A_v} \sum_v A_v - J_{A_c} \sum_c A_c \\ & - J_{X_e} \sum_e X_e - J_{K_e} \sum_e K_e - J_{B_e} \sum_e B_e \\ & - J_{Z_f} \sum_f Z_f - J_{K_f} \sum_f K_f - J_{B_f} \sum_f B_f, \end{aligned} \quad (52)$$

which has fewer tunable parameters. This simplification means that we might not find all of the phases in the full Hamiltonian (51), but we can still find several interesting phases. As the A_v and A_c terms commute with everything else, we can choose to set $J_{A_v} = J_{A_c} = \infty$. This in turn is equivalent to enforcing a symmetry. In fact, A_v and A_c each generate a 1-form symmetry, so we realize we are studying 1-form symmetry-protected topological (SPT) phases.

To find some phases of H_{ITC} , tune to a limit where some J_G are much larger than others. If the largest terms all commute, the resulting Hamiltonian is exactly solvable. For example, if the coefficients of the single-site terms J_{X_e} and J_{Z_f} are much larger than the rest of the check coefficients, then the effective

Hamiltonian is the paramagnet Hamiltonian

$$H_{\text{Para}} = -J_{X_e} \sum_e X_e - J_{Z_f} \sum_f Z_f, \quad (53)$$

with some small perturbations. Some other exactly solvable Hamiltonians are

$$H_{\text{TCE}} = -J_{A_v} \sum_v A_v - J_{Z_f} \sum_f Z_f - J_{B_f} \sum_f B_f, \quad (54)$$

$$H_{\text{TCF}} = -J_{A_c} \sum_c A_c - J_{X_e} \sum_e X_e - J_{B_e} \sum_e B_e, \quad (55)$$

which each represent a single copy of the toric code stacked with a paramagnet, and

$$\begin{aligned} H_{2\text{TC}} = & -J_{A_v} \sum_v A_v - J_{A_c} \sum_c A_c \\ & - J_{B_e} \sum_e B_e - J_{B_f} \sum_f B_f, \end{aligned} \quad (56)$$

which represents the two toric codes from (6). Some distinct choices result in the same phases. For example, the Hamiltonian

$$H = -J_{A_v} \sum_v A_v - J_{Z_f} \sum_f Z_f - J_{K_f} \sum_f K_f \quad (57)$$

belongs to the same phase as H_{TCE} because $Z_f K_f = B_f$.

Another useful limit is the RBH Hamiltonian

$$H_{\text{RBH}} = -J_{K_e} \sum_e K_e - J_{K_f} \sum_f K_f, \quad (58)$$

whose ground state is the 3D cluster state [13]. This Hamiltonian is in the trivial phase, but in a nontrivial SPT phase under the 1-form symmetry generated by A_v and A_c . Altogether, we find five phases in \mathcal{H}_{ITC} , represented by the five named Hamiltonians above.

Some of the exactly solvable Hamiltonians have previously been shown to lead to symmetry-protected self-correction. These are H_{RBH} [14], H_{Para} [15], and $H_{2\text{TC}}$ [16]. In each case the protecting symmetry is (possibly a subset of) the 1-form symmetry generated by A_v and A_c . However, in all three cases the boundary conditions must be chosen differently to lead to symmetry-protected self-correction rather than single-shot error correction. It is likely that the TCE and TCF phases would also support self-correction with the right choice of boundary and 1-form symmetry. In addition, the RBH phase supports symmetry-protected topological order at nonzero temperatures [17], and the other phases likely do as well.

B. Topological order in each phase

Every phase that Ref. [7] finds in the phase diagram of the KVC has topological order somewhere, either in the bulk or on the boundary. In the ITC, the phases corresponding to H_{TCE} , H_{TCF} , and $H_{2\text{TC}}$ are topologically ordered in the bulk. In the paramagnetic and RBH phases, the topological order instead must occur on the boundary. The symmetry and the bulk Hamiltonian together restrict the possible gapped boundaries, as is common in SPTs. For example, consider the paramagnetic phase. On the trivial boundary, we can include an X_e term for every boundary edge without causing problems,

giving paramagnetic boundary conditions. On the intertwined boundary, however, such a term is not in \mathcal{G}_{ITC} and would anticommute with the stabilizers. Instead, we must include the terms

$$A_v^{2d} = \text{green cross}, \quad B_f^{2d} = \text{blue parallelogram}, \quad (59)$$

which make up a 2D toric code on this boundary. We can check that these terms are in \mathcal{G}_{ITC} . Similarly, in the RBH phase, we can include K_e for edges in the intertwined boundary, leading to no topological order there. The trivial boundary must instead have the terms

$$A_v^{2d, \text{eff}} = \text{green cross on green plane}, \quad B_f^{2d} = \text{blue parallelogram}, \quad (60)$$

which define an effective 2D toric code there. These terms are in \mathcal{G}_{ITC} and define the standard toric code boundary conditions of the RBH Hamiltonian.

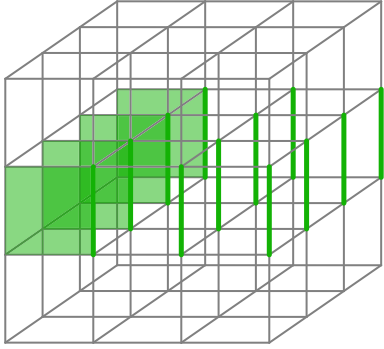
In the TCE and 2TC phases the boundary conditions are just the “smooth” boundaries of the 3D toric codes, where membrane operators may terminate but stringlike operators may not. In the TCF phase this is still the case, but it is less clear on the intertwined boundary. If we include the K_e operators on the boundary edges, then the naive termination of A_c does not commute with these. Instead, we use the dressed termination of K_e from (12) and A_c from (13), so that the terms in the Hamiltonian are

$$\text{blue cube}, \quad \text{green parallelogram}, \quad \text{green square}, \quad (61)$$

along with the terms in the bulk and on the trivial boundary. This new boundary looks funny, but is still a valid (smooth) termination of the 3D toric code on faces. It is related to the ordinary termination (on the trivial boundary) by the U_{CX} symmetry (9).

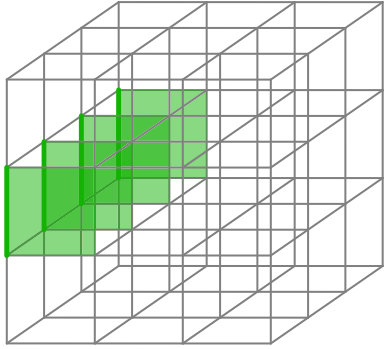
Reference [7] additionally shows that the bare logical operators in the subsystem code act as logical operators in every phase. In the ITC, this is particularly straightforward in the paramagnetic phase; \bar{X} (16) and \bar{Z} (17) consist of the boundary X -type logical operator dressed by X_e terms and the boundary Z -type logical operator dressed by Z_f terms, respectively. In the paramagnetic phase, the single-site X_e and Z_f terms are stabilizers, so \bar{X} and \bar{Z} remain logical operators.

In the RBH phase the relationship is not as obvious. To see that it still holds, consider a product of X -type K_e stabilizers,



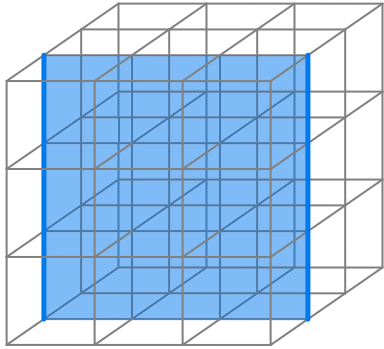
(62)

and an X logical operator on the boundary toric code,



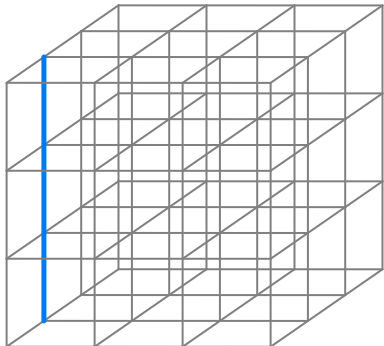
(63)

recalling that both figures are meant to be periodic in the front/back and top/bottom directions. Multiplying these together does in fact give \tilde{X} , as in (16). Similarly a product of Z -type K_f stabilizers,



(64)

and a Z logical operator,



multiply \tilde{Z} , as in (17).

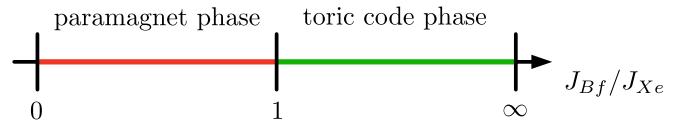


FIG. 5. Phase diagram for the Hamiltonian in (66). Within the A_v - and Z_f -satisfying Hilbert space, the B_f and X_e terms are dual to each other, so that the phase transition occurs when the couple strengths are equal. The system enters the 3D toric code phase when J_{Bf} is large and enters the paramagnetic phase when J_{Xe} is larger.

In the TCE phase, \tilde{X} is the membrane logical operator and \tilde{Z} is the stringlike logical operator dressed with single-site Z_f stabilizers. In the TCF phase, \tilde{Z} is the membrane logical operator, whose dressing is required as in (61). The single-site X_e operators in \tilde{X} are all individual stabilizers except for those on the intertwined boundary. The latter can be multiplied by K_e on every edge to give the stringlike logical operator in the TCF phase. Since the 2TC phase just consists of the toric code on edges stacked with the toric code on faces, it is clear that \tilde{X} acts only on one code while \tilde{Z} acts on both. Thus, one of the logical qubits in the 2TC phase is privileged over the other. It would be interesting to find a more systematic understanding of the relationship between logical qubits in the parent subsystem code and logical qubits in individual gapped phases.

C. Constructing phase diagrams

So far, we have been analyzing individual phases via exactly solvable Hamiltonians. By including noncommuting terms with comparable strength, we can instead build Hamiltonians with phase transitions. For example, the Hamiltonian

$$H = -J_{Av} \sum_v A_v - J_{Zf} \sum_f Z_f - J_{Bf} \sum_f B_f - J_{Xe} \sum_e X_e \quad (66)$$

describes 3D \mathbb{Z}_2 lattice gauge theory; the first two terms commute with everything and are always satisfied, but the second two terms compete. Within the space of states satisfying the first two terms, the second two terms are dual [18,19], so that the phase transition happens at $J_{Bf} = J_{Xe}$ [20]. When $J_{Bf} > J_{Xe}$ the Hamiltonian is in the TCE phase, while when $J_{Bf} < J_{Xe}$ the Hamiltonian is in the paramagnetic phase, as shown in Fig. 5.

Including more terms, as in

$$H = -J_{Av} \sum_v A_v - J_{Ac} \sum_c A_c - J_{Zf} \sum_f Z_f - J_{Xe} \sum_e X_e - J_{Be} \sum_e B_e - J_{Bf} \sum_f B_f, \quad (67)$$

results in a larger phase diagram. Note that each term acts only on edges or only on faces, so that we can analyze each sector separately. The phases and phase transitions are shown in Fig. 6. The phase transitions happen at $J_{Bf} = J_{Xe}$ and $J_{Be} = J_{Zf}$.

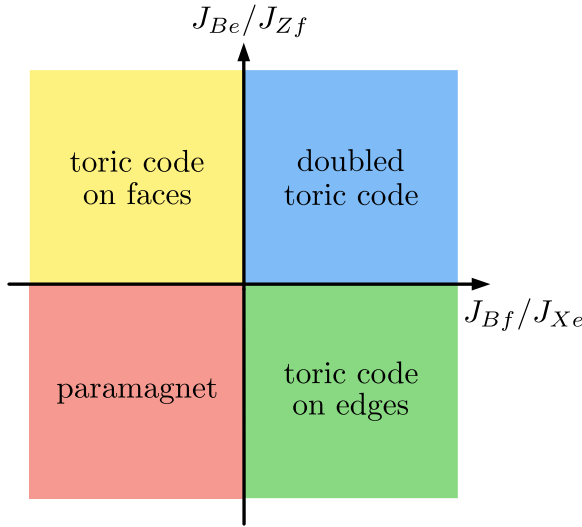


FIG. 6. Phase diagram for the Hamiltonian in (67). The four phases—paramagnetic, toric code on edges, toric code on faces, and doubled toric code—have fixed points described by (53), (54), (55), and (56), respectively. The phase boundaries are vertical and horizontal lines because the edge qubits and face qubits are decoupled in the bulk.

Another useful Hamiltonian is

$$H = -J_{Av} \sum_v A_v - J_{Ac} \sum_c A_c - J_{Zf} \sum_f Z_f - J_{Xe} \sum_e X_e - J_{Ke} \sum_e K_e - J_{Kf} \sum_f K_f, \quad (68)$$

which is the same as (67) but with the B terms replaced with K terms. Using duality transformations like those in Ref. [21], we can show that the phase transitions happen at $J_{Ke} = J_{Zf}$ and at $J_{Kf} = J_{Xe}$. The resulting phase diagram looks like Fig. 7. The full Hamiltonian H_{ITC} has a higher-dimensional phase diagram, with each of the five described phases appearing.

The analysis of the phase diagram suggests some possible connections to the KVC. Reference [7] shows that the KVC supports four phases: two 3D toric code phases and two trivial phases, where the trivial phases have 2D toric codes on opposite boundaries. In the ITC, the paramagnetic and RBH phases are both trivial in the bulk, in the sense that neither is topologically ordered. However, they cannot be continuously connected without closing the gap or breaking the symmetry, meaning that H_{RBH} is in an SPT phase. Thus, we say that they are not the same phase but that both are trivial. Note that they have their 2D toric codes on opposite boundaries, as is true of the two trivial phase in Ref. [7]. In this sense, H_{para} , H_{RBH} , H_{TCE} , and H_{TCF} are all found in the KVC. These phases arrange into a phase diagram (Fig. 6 of Ref. [7]) that matches Fig. 7 in the ITC.

However, the ITC also includes the doubled toric code phase (H_{2TC}). Reference [7] did not find such a phase in the KVC, but they did not exhaustively search the entire parameter space. If the KVC also contains a doubled toric code phase, then it would be possible that the KVC and ITC are related by

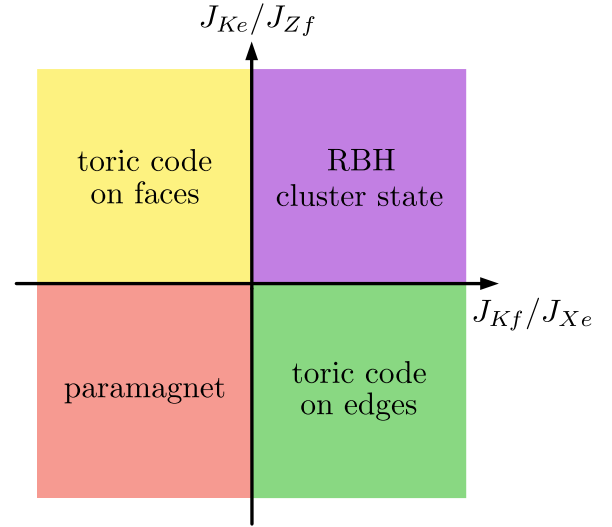


FIG. 7. Phase diagram for the Hamiltonian in (68). The four phases—paramagnetic, toric code on edges, toric code on faces, and RBH cluster state—have fixed points described by (53), (54), (55), and (58), respectively. Analysis similar to Ref. [21] allows us to infer that the phase boundaries are vertical and horizontal lines. Compare to Fig. 6 of Ref. [7].

some finite-depth unitary circuit, and that they are equivalent as subsystem codes.

V. REMAINING QUESTIONS

In this paper we have introduced a single-shot quantum error-correcting code, the intertwined toric code. The ITC possesses three main advantages. First, the ITC is physically motivated in that it descends directly from underlying toric codes and that the procedure that builds the ITC confines the problematic pointlike excitations of the toric codes. Second, the resulting code has geometrically simple logical operators and decoding procedures, both descending from the geometric simplicity of the toric codes. Third, the transparent nature of the check operators makes extracting the phase diagram for the ITC straightforward.

This work uncovers a number of promising avenues for future research. In Sec. II we constructed the ITC by adding single-site check operators to 3D toric codes, while in Sec. III we instead constructed it by adding single-site check operators to the RBH Hamiltonian. Both constructions suggest a generalization to arbitrary Walker-Wang models. The Walker-Wang construction [22] takes a 2D anyon theory as input and outputs a 3D lattice model. Using two transparent bosons as the input theory gives two copies of the toric code [23] while using the 2D toric code as input gives the RBH Hamiltonian [24]. In both cases, the vertex terms of the Walker-Wang model are equivalent to the stabilizers in the ITC. This suggests a procedure for turning a 2D anyon theory into a 3D subsystem code: Construct the Walker-Wang model corresponding to the anyon theory and then supplement with single-site check operators so that the vertex terms remain stabilizers but the face terms become check operators. Reference [25] defines another procedure for constructing 3D subsystem codes from 2D anyon theories. How are these two procedures related?

Could the subsystem Walker-Wang construction shed light on the differences between the GCC and the codes constructed in Ref. [25]?

The RBH Hamiltonian also provides for fault-tolerant measurement-based quantum computation [26], wherein a 2D toric-code degree of freedom is teleported across a 3D lattice system. Initially, the bulk of the system is in the ground state of the RBH Hamiltonian and the 2D toric code is on the left boundary. After measuring single-qubit operators in the bulk, the toric code ends up on the right boundary. In the language of the ITC, this process consists of using check operators to move from the paramagnetic phase to the RBH phase. In fact, single-shot error-correction procedures in the KVC correspond to scheduled phase transitions as well [7]. What conclusions can be drawn from this connection between single-shot error correction and fault-tolerant measurement-based quantum computation?

One of the advantages of the KVC is that each check operator acts on at most three qubits. Is it possible to rewrite the ITC so that no check operator acts on more than three qubits? The presentation in Sec. II uses smaller check operators. Is it possible to perform single-shot error correction using measurements of these check operators? Heuristically, it seems possible because a single measurement error introduces a geometrically close pair of violated stabilizers, rather than a single isolated violated stabilizer (as in the 2D toric code). However, this claim would have to be checked.

The boundaries of the ITC (and the GCC and the KVC) hold outsized importance. As originally introduced in Sec. II B, the ITC with no boundaries encodes no logical qubits. However, the analysis of Sec. III shows that we can reliably detect syndromes and correct errors even without a boundary. Taken together, these statements show that reliable error correction, which follows from confinement in the ITC, is insufficient for single-shot error correction of a logical qubit. What, then, is the physical interpretation of the boundary? Furthermore, the bare logical operators commute in the bulk and only anticommute on the boundary. In the phase diagrams of Sec. IV, the boundaries “host” the topological order when there is no topological order in the bulk (in the paramagnetic and RBH phases). The boundaries also ruin self-correction in the sense that models exist that are self-correcting in the presence of generic perturbations in the bulk but only with symmetry-allowed perturbations on the boundary [16]. Clearly, the relationship between the bulk and the boundary contains interesting unanswered questions.

Lastly, all known single-shot stabilizer codes are self-correcting (but exist only in four dimensions or higher). None of the phases that we found in the ITC phase diagram are self-correcting. Can the simplified construction of the ITC offer any progress towards a new subsystem code that does include a self-correcting phase in three dimensions?

ACKNOWLEDGMENTS

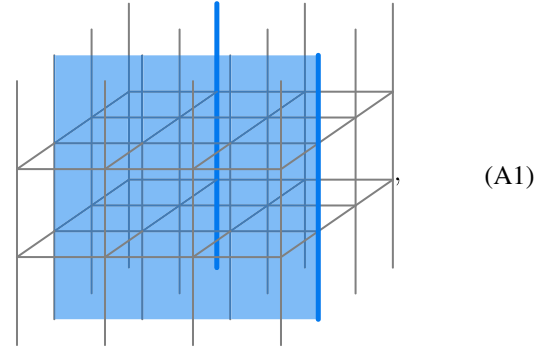
I would like to thank Rahul Nandkishore for continued support and direction during this process. I also thank Yaodong Li and Marvin Qi for helpful discussion about the project,

and Tyler Ellison, Aaron Friedman, Oliver Hart, Sam Roberts, David T. Stephen, and Matteo Wilczak for feedback on the manuscript. This work was supported by the U.S. Department of Energy, Office of Science, Basic Energy Sciences, under Award No. DE-SC0021346.

APPENDIX: MORE BOUNDARY CONDITIONS

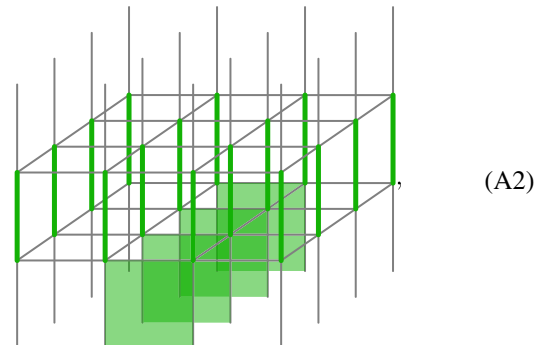
As promised, we here define the e -condensed and m -condensed boundary conditions so that the ITC can be defined with a single logical qubit on a cubic lattice that is itself a cube (with no periodic boundary conditions).

We put the e -condensed boundary on the top and bottom of the cube. To do so, start with the previously defined trivial boundary and remove qubits from the boundary edges. We are left with the same B_e check operators but some B_f check operators are truncated to three-body operators, which we leave in the check group. Now there are no boundary A_v stabilizers, but we keep the five-body A_c stabilizers. On this boundary, the bare logical membrane operator \bar{Z} and the dressed logical stringlike operator \bar{Z}' may both terminate,



but the X -type logical operators may not.

To construct the m -condensed boundary on the front and back of the cube, once again start with the trivial boundary. Now, add qubits to every boundary face. For every boundary edge we can now define a three-body B_e operator. We also define single-body Z_f operators on these new face qubits. The resulting stabilizers are the five-body A_v operators we already had and new six-body $A_c = \prod_{f \in \partial c} Z_f$ operators. The bare logical membrane operator \bar{X} and the dressed logical stringlike operator \bar{X}' may both end on this boundary,



but the Z -type logical operators may not.

These boundaries result in a cleaner code because they encode only a single logical qubit. There are $3L^3 + 2L^2 - L$ edges and $3L^3 - L^2$ faces for a total of $6L^3 + L^2 - L$ physical qubits. On these qubits we enforce $3L^3 - 3L$ single-site X_e check operators, $2L^2 + 2L$ of the K_e check operators (counting only those on the intertwined boundary), and $3L^3 - L^2$ single-site Z_f check operators. Between $3L^3 + L^2$ total B_f check operators we have L^3 relations and therefore $2L^3 + L^2$ independent B_f operators. There are $3L^3 - 2L^2 - L$ total B_e check operators with $L^3 - L^2 - L + 1$ relations between them, giving $2L^3 - L^2 - 1$ independent B_e operators. Along with $L^3 + L^2 - L - 1$ of the A_v stabilizers and L^3 of the A_c ,

we have

$$K = N - \frac{1}{2}(\log_2 |\mathcal{G}| + \log_2 |\mathcal{S}|) = 1 \quad (\text{A3})$$

logical qubit.

We should think of this as the minimal intertwined toric code because it encodes only a single logical qubit and has no need for periodic boundary conditions. We can go through the analysis of Sec. IV to see what these boundaries become in the different phases. For example, the e -condensed boundary becomes the rough boundary conditions for the 2D toric code in the paramagnetic phase and for the 3D toric code in the TCE phase.

-
- [1] H. Bombín, Single-shot fault-tolerant quantum error correction, *Phys. Rev. X* **5**, 031043 (2015).
 - [2] A. Yu. Kitaev, Fault-tolerant quantum computation by anyons, *Ann. Phys.* **303**, 2 (2003).
 - [3] E. Dennis, A. Yu. Kitaev, A. Landahl, and J. Preskill, Topological quantum memory, *J. Math. Phys.* **43**, 4452 (2002).
 - [4] H. Bombín, Gauge color codes: Optimal transversal gates and gauge fixing in topological stabilizer codes, *New J. Phys.* **17**, 083002 (2015).
 - [5] A. Kubica and M. Vasmer, Single-shot quantum error correction with the three-dimensional subsystem toric code, *Nat. Commun.* **13**, 6272 (2022).
 - [6] A. Kubica and B. Yoshida, Ungauging quantum error-correcting codes, [arXiv:1805.01836](https://arxiv.org/abs/1805.01836).
 - [7] Y. Li, C. W. von Keyserlingk, G. Zhu, and T. Jochym-O'Connor, Phase diagram of the three-dimensional subsystem toric code, [arXiv:2305.06389](https://arxiv.org/abs/2305.06389).
 - [8] M. B. Hastings and J. Haah, Dynamically generated logical qubits, *Quantum* **5**, 564 (2021).
 - [9] D. Poulin, Stabilizer formalism for operator quantum error correction, *Phys. Rev. Lett.* **95**, 230504 (2005).
 - [10] S. Bravyi, G. Duclos-Cianci, D. Poulin, and M. Suchara, Subsystem surface codes with three-qubit check operators, *QIC* **13**, 963 (2013).
 - [11] T. D. Ellison, Y.-A. Chen, A. Dua, W. Shirley, N. Tantivasadakarn, and D. J. Williamson, Pauli topological subsystem codes from abelian anyon theories, *Quantum* **7**, 1137 (2023).
 - [12] C. Castelnovo and C. Chamon, Topological order in a three-dimensional toric code at finite temperature, *Phys. Rev. B* **78**, 155120 (2008).
 - [13] R. Raussendorf, S. Bravyi, and J. Harrington, Long-range quantum entanglement in noisy cluster states, *Phys. Rev. A* **71**, 062313 (2005).
 - [14] S. Roberts and S. D. Bartlett, Symmetry-protected self-correcting quantum memories, *Phys. Rev. X* **10**, 031041 (2020).
 - [15] C. Stahl and R. Nandkishore, Symmetry-protected self-correcting quantum memory in three space dimensions, *Phys. Rev. B* **103**, 235112 (2021).
 - [16] C. Stahl, Self-correction from higher-form symmetry protection on a boundary, *PRX Quantum* **4**, 030341 (2023).
 - [17] S. Roberts, B. Yoshida, A. Kubica, and S. D. Bartlett, Symmetry-protected topological order at nonzero temperature, *Phys. Rev. A* **96**, 022306 (2017).
 - [18] E. Fradkin and L. Susskind, Order and disorder in gauge systems and magnets, *Phys. Rev. D* **17**, 2637 (1978).
 - [19] F. J. Wegner, Duality in generalized Ising models and phase transitions without local order parameters, *J. Math. Phys.* **12**, 2259 (1971).
 - [20] M. Creutz, L. Jacobs, and C. Rebbi, Experiments with a gauge-invariant Ising system, *Phys. Rev. Lett.* **42**, 1390 (1979).
 - [21] M. Mühlhauser, K. P. Schmidt, J. Vidal, and M. R. Walther, Competing topological orders in three dimensions, *SciPost Phys.* **12**, 069 (2022).
 - [22] K. Walker and Z. Wang, (3 + 1)-TQFTs and topological insulators, *Front. Phys.* **7**, 150 (2012).
 - [23] C. W. von Keyserlingk, F. J. Burnell, and S. H. Simon, Three-dimensional topological lattice models with surface anyons, *Phys. Rev. B* **87**, 045107 (2013).
 - [24] S. Roberts and D. J. Williamson, 3-Fermion topological quantum computation *PRX Quantum* **5**, 010315 (2024).
 - [25] J. C. Bridgeman, A. Kubica, and M. Vasmer, Lifting topological codes: Three-dimensional subsystem codes from two-dimensional anyon models, *PRX Quantum* **5**, 020310 (2024).
 - [26] R. Raussendorf, J. Harrington, and K. Goyal, A fault-tolerant one-way quantum computer, *Ann. Phys.* **321**, 2242 (2006).