The two-dimensional color code is an alternative to the toric code that encodes more logical qubits while maintaining crucial features of the $\mathbb{Z}_2 \times \mathbb{Z}_2$ toric code in the long wavelength limit. However, its short range physics include single qubit Pauli operations that violate either three or six stabilizers as opposed to the toric code where single qubit Pauli operations violate two or four stabilizers. Exploiting this fact we construct several non-CSS versions of the two-dimensional color code falling into two families - those where either three, four or five stabilizers are violated and those which violate exactly four stabilizers for all the three types of single qubit Pauli operations. These models are not equivalent to the original color code by a local unitary transformation. Nevertheless the code spaces of the CSS and non-CSS versions are related by local unitaries which reflects the fact that their long range physics coincide. This also implies that the non-CSS versions host transversal Clifford gates and hence support fault-tolerant computations. As a consequence of the non-CSS structure, the logical operators are of a mixed type which in some cases include all the three Pauli operators making them potentially useful for protection against biased Pauli noise.
I. INTRODUCTION

One of the major challenges in building quantum computers is detecting and correcting errors in the form of noise and decoherence that could occur during information processing. In the quest to move from the ‘noisy-intermediate-scale-quantum era’ to full-fledged ‘fault-tolerant application-scale-quantum era’, we need to deploy quantum error correcting codes (QECC) [1]. The role of QECC is to reduce the errors in the qubits that carry the information by adding redundancies. The way this is achieved is using multiple physical qubits to encode a single logical qubit using a suitable code, such that the logical qubit has lower error rates compared to the underlying physical qubit.

A good candidate to overcome this is by using the method of fault tolerant quantum computing, which are realized via many-body systems with topological order, a prominent example being the toric code [2] also known as the surface codes. These models can be thought of as a Hamiltonian realization of planar gauge theories [3, 4] based on discrete, finite groups and more generally known as the quantum double models [5, 6]. Following Kitaev’s work another Hamiltonian realizing the abelian \( \mathbb{Z}_2 \times \mathbb{Z}_2 \) topological phase appeared in the form of the color code [7] which is not written in the form of a lattice gauge theory. Instead, in these models the degrees of freedom or the physical qubits are located on the vertices of a trivalent and tricolorable lattice. The standard examples of such lattices include the hexagonal lattice (6,6,6), the square-octagonal lattice (4,8,8) and a (4,6,12) lattice. While these are regular lattices with translational invariance, it is possible to construct several other irregular lattices on which the color code is well defined. A remarkable property of the color code is that it encodes more number of qubits than the toric code on the same space, supports fault tolerant computation as they host transversal gates and is more efficient for quantum error correction [8, 9]. The simplest example of the color code is the one defined on a triangle with 7-qubits and this coincides with the 7-qubit Steane code [10–15], a well known example of a fault tolerant code.

The color code has seen several interesting generalizations [16–26] and has also been experimentally realized [27–36]. Apart from direct generalizations several theoretical works have explored the physics and error correction properties of the color code and related models [37–56]. Connections between toric and color codes were presented in [57–61].

Both the surface codes and the color codes are examples of Calderbank-Steane-Shor (CSS) codes as each of the stabilizers are entirely composed of either the Pauli, \( x \) or \( z \) operators. It is of interest to consider non-CSS generalizations as they are thought to be of particular interest in biased noise settings [62–65] where one Pauli error is more likely than the other. In this paper we take a step in this direction by presenting several non-CSS color codes on 2D trivalent, tricolorable lattices. We provide a generic method to construct abelian stabilizer codes on trivalent lattices that are naturally non-CSS. We find that these models are not equivalent to the original color code by a local unitary (LU) transformation. This is de-
spite the fact that the non-CSS versions share the same topological properties as the original color code, that is they continue to realize the $\mathbb{Z}_2 \times \mathbb{Z}_2$ topological phase. By this we mean that the code spaces of the two models and the anyonic excitations are LU related. This also implies that the non-CSS versions continue to support fault tolerant quantum computation. However the original color code has more excitations that are local and immobile, that cannot be moved around like the anyons. The non-CSS versions have different kinds of immobile excitations when compared to the original color code. They differ in the fact that their energies are not the same as their CSS counterparts. This serves as the starting point of the generalizations and they naturally lead to non-CSS versions of the color code.

The paper is organized as follows. We begin with an overview of the original CSS color code in Sec. II. We go over some of its topological properties, including the computation of its ground state degeneracy on genus, $g$ surface and their anyon content. We also look at the immobile excitations of the theory through the application of Pauli operators on an individual qubit and call these the elementary excitations. The anyons can be obtained by combining these elementary excitations. The construction of the non-CSS stabilizer codes on general trivalent lattice starts in Sec. IV. We take a bottom-up approach here by looking at operator configurations around different types of vertices on a trivalent lattice and at operator configurations along edges. These building blocks are used to construct the full stabilizer codes on hexagonal lattices in Sec. V. We discuss three types of non-CSS color codes, the [444]-color codes, the deconfined [345]-color codes and the confined [345]-color codes. The notation is explained in the text. In this language the original color code becomes the [336]-color codes. We present all their non-CSS versions for completion. The topological properties of each of these models is discussed in Sec. VI. The finite versions of these codes, namely the triangle codes are studied in Sec. VII. We thoroughly analyze one example of a fully mixed non-CSS code in Sec. VIII to study the mixed nature of the string logicals in these models. In Sec. IX we construct trivalent and tricolorable lattices on higher genus surfaces and build the non-CSS codes on them. We end with an outlook in Sec. X. There are several appendices exploring more models. We present these for completion.

II. PRELIMINARIES

To begin with we go over the stabilizer formalism and the topological properties of the canonical color code as originally proposed [7]. We also study the unitary operations that permute Pauli matrices which will be used later in the paper.

A. Stabilizer formalism and commuting projector codes

Stabilizer codes provide a convenient framework to study QECCs [8]. We briefly recall some of the basic concepts of the stabilizer framework that we will use in the rest of the article. Let $\mathcal{P}_n$ denote the Pauli group on $n$ qubits. The Pauli group is defined as the group generated by the following operators

$$\mathcal{P}_n = \langle X, Y, Z, \pm \mathbb{I}, \pm i \mathbb{I} \rangle^\otimes n,$$

i.e., its the group generated by $n$-strings of single qubit Pauli operators with $\omega^4 = 1$. We will only be concerned with stabilizer codes associated with the Pauli group in this article. A stabilizer code is described by a stabilizer group $\mathcal{S}$ which is an abelian subgroup of the Pauli group that is generated by stabilizers (commuting operators) $s$ where

$$\mathcal{S} = \{ s \in \mathcal{P}_n \mid [s_i, s_j] = 0 \},$$

and also $-\mathbb{I}$ is not in the stabilizer set. Let us denote by the centralizer of $\mathcal{S}$ by $\mathcal{Z}$, which consists of the elements of $\mathcal{P}$ that commute with all the elements of $\mathcal{S}$. The code space $\mathcal{C}$ is defined as the subspace of the $n$-qubit Hilbert space stabilized by $\mathcal{S}$, where

$$|\psi\rangle \in \mathcal{C} \text{ if } s|\psi\rangle = +1|\psi\rangle, \quad \forall s \in \mathcal{S}.$$  

Then the set of logical operators of the code belong to the set $\mathcal{Z} - \mathcal{S}$. Note that these elements leave the code space invariant, however, they would apply some non-trivial operation on the individual code words, that would be undetected by the syndromes of the stabilizers.

So we see that the stabilizer code is completely specified by the stabilizers, and the underlying group structure gives rise to the specifics of the code. The stabilizer Hamiltonian is just defined as

$$H(\mathcal{S}) = -\sum_j |\mathcal{S}| s_j.$$  

Note that code words are the ground states of $H(\mathcal{S})$ since all the stabilizers account for $-1$ energy in the Hamiltonian.

There is an alternative representation for stabilizer code Hamiltonians into what are known as commuting projector code (CPC) Hamiltonians. A CPC Hamiltonian has terms made of projectors $\mathbb{P} = \langle P_j \rangle$ such that $[P_i, P_j] = 0$, and

$$H(\mathbb{P}) = -\sum_{j=1}^{\mid\mathbb{P}\mid} P_j.$$  

There are no other constraints of CPC Hamiltonians. It is straightforward to recast any Pauli stabilizer code Hamiltonian as a CPC by defining

$$P_j = \frac{1}{2}[\mathbb{I} + s_j].$$
Note that in this formulation the projector onto the code space \( P_C \) is given by
\[
P_C = \prod_{j=1}^{\vert F \vert} P_j. \tag{7}
\]

For topological codes defined on finite lattices, the dimension of the code space is \( \text{dim}C = 2^{n-|S|} \). Note that the dimension of the code space is also known as the ground state degeneracy (GSD) of the corresponding stabilizer code or CPC Hamiltonian. An alternate method to obtain \( \text{dim}C \) is to compute the trace of \( P_C \). For codes defined on a closed manifold of a given genus, one would need to use the Euler formula for such a counting method to determine \( \text{dim}C \). We will deploy different methods depending on the context.

**B. Overview of the canonical color code**

Two-dimensional color codes can be cast as commuting projector Hamiltonians in a manner similar to abelian toric codes or quantum double Hamiltonians [2, 5, 6]. The models are defined on a trivalent and tricolorable lattice discretizing a two-dimensional surface. An example is as shown in Fig. 1, where the physical qubits are placed on the vertices (sites) of the lattice.

![Figure 1. A hexagonal lattice as an example of a trivalent, tricolorable lattice supporting the canonical color code.](image)

The projectors are defined for the hexagons of each color and are built out of the stabilizers, as
\[
PX_j = \frac{1}{2} (1 + X_j), \tag{8}
\]
and similar expressions for the other two colors. The operators \( X_j \), for each color, are the Pauli \( X \) stabilizers as depicted in Fig. 2, with \( X_j = \prod_{v_j=1}^{6} x_{v_j} \), where \( v_j \) are the vertices filling up the face \(( f )\) labelled by the subscript \( j \). The operator 1 is shorthand for \( \prod_{j=1}^{6} 1_{v_j} \). Analogous projectors are defined for the \( Z \) stabilisers by replacing \( x \) with \( z \) in Eq. (8).

![Figure 2. The canonical color code stabilizers depicted in a unit cell of the trivalent and tricolorable lattice. The figures in the left and right panel denote the two types of stabilizers associated with each colored face.](image)

Using these operators we write down the commuting projector Hamiltonian as
\[
H = - \sum_{j=1}^{f} (PX_j + PZ_j) - \sum_{j=1}^{f} (PX_j + PZ_j) - \sum_{j=1}^{f} (PX_j + PZ_j), \tag{9}
\]
where \( f \), \( f \) and \( f \) are the number of red, blue and green faces respectively. For a hexagonal lattice they all equal \( \frac{|F|}{3} \), but for an arbitrary trivalent and tricolorable lattice we have, \( f + f + f = |F| \).

1. **Ground State Degeneracy**

The model in Eq. (9) is exactly solvable as it is easy to verify that all the terms commute with each other. On a surface with genus \( g \), the model has a ground state degeneracy (GSD) given by \( 2^{4g} \). We compute the GSD as
\[
\text{GSD} = \text{Tr} \left[ \prod_{j=1}^{f} P_{X_j} P_{Z_j} \prod_{j=1}^{f} P_{X_j} P_{Z_j} \prod_{j=1}^{f} P_{X_j} P_{Z_j} \right], \tag{10}
\]
which is essentially the trace of the projector to the ground state space of the color code Hamiltonian, Eq. (9). The product consists of several tensor products of Pauli operators having support on different sites, however these terms are traceless and hence do not contribute to the GSD. The only terms contributing to the trace are those of the form \( \prod_{j=1}^{f} 1_{v_j} \) with support on all sites. Thus this GSD computation is an exercise in counting the number of such identity terms.

Expanding each of the products of projectors in the
stabilizers on a state left invariant by the Z-type stabilizers can be constructed by applying only the X-type projectors encoding 4 logical qubits is precisely $2^{2g}$.

From these expressions we can see that the projector to the ground state manifold has precisely $2^{2g}$ number of identity operator on the entire lattice. These are accounted for by using the following constraints on the stabilisers making up the Hamiltonian defined on a closed manifold of genus $g$.

$$\prod_{j=1}^{f} PX_j = \frac{1}{27} \left[ 1 + \prod_{j=1}^{f} X_j \right] + \text{traceless terms},$$

$$\prod_{j=1}^{f} PZ_j = \frac{1}{27} \left[ 1 + \prod_{j=1}^{f} Z_j \right] + \text{traceless terms},$$

and similar expressions for the other colored faces.

We can thus evaluate the GSD as

$$\text{GSD} = \frac{1}{27} \frac{1}{27} \frac{1}{27} 2^{V/2} 2^f.$$  

Using $f+f+f = |F|$ the exponent becomes

$$|V| - 2|F| + 4 = 4g,$$

where we have used the Euler formula, $|V| - |E| + |F| = 2 - 2g$ and the fact that $|E| = \frac{3|V|}{2}$ for a trivalent and tricolorable lattice. This completes the first proof for the GSD of the canonical color code on a closed manifold of genus $g$.

The more conventional proof found in the literature involves finding the number of independent stabilisers in the Hamiltonian, Eq.(9). From the six constraint relations in Eq.(12) we deduce that only four of them are independent, resulting in $2|F| - 4$ independent stabilisers for this system. Thus the number of encoded qubits is precisely $|V| - 2|F| + 4$ which evaluates to $4g$ as shown above.

We will mostly employ the second method to compute the code space dimension of the non-CSS color codes considered in this paper.

2. Code space $C$ and logical operators

The Hamiltonian in Eq.(9) is a sum of commuting projectors encoding $4g$ logical qubits. One of the ground states can be constructed by applying only the X-type stabilizers on a state left invariant by the Z-type stabilizers,

$$|0_1, \cdots, 0_{4g} \rangle = N \prod_{j=1}^{f} PX_j \prod_{j=1}^{f} PX_j \prod_{j=1}^{f} PX_j |s\rangle,$$

where $g$ denotes the genus of the surface and $|s\rangle$ is a state stabilised by the projectors $PZ_j, PZ_j, PZ_j$ for all $j \in \{1, \cdots, f\}$, $j \in \{1, \cdots, f\}$ and $j \in \{1, \cdots, f\}$ respectively. The $0_j$ denotes the $j$th logical zero state and $N$ is the overall normalization of the state.

To construct the other ground states or the code words we identify the winding operators along the non-contractible loops of the genus $g$ manifold. For each non-contractible loop, $C_{\alpha,j}$, where $\alpha \in \{1, 2, \cdots, g\}$ enumerates the different holes of the surface and $j \in \{1, 2\}$ provides the index for the two loops for a given hole. This is because we have two sets of winding operator along two chosen colors as shown in Fig. 3. We pick the winding operators along the red and blue shrunk lattices of the color code.

![Figure 3. The winding operators on the red and blue shrunk lattices of the color code.](image-url)

There is no loss of generality in this choice as an operator along a given shrunk lattice is obtained as a product of operators along the other two shrunk lattices as shown in Fig. 4. In other words, only two of the three colors are independent with respect to the winding or logical operators. The winding operators comprising the logical Pauli $X$ operators are given by

$$\hat{X}_{\alpha,j} = \prod_{i \in C_{\alpha,j}} x_i; \quad \hat{X}_{\alpha,j} = \prod_{i \in C_{\alpha,j}} x_i,$$

where $\alpha = 1, 2, \cdots, g$ and $j = 1, 2$.
with $\alpha \in \{1, 2, \cdots , g\}$ and $j \in \{1, 2\}$ along the red and blue shrunk lattices respectively. The logical Pauli $Z$ operators are analogously written as

$$Z_{\alpha,j} = \prod_{i \in C_{\alpha,j}} z_i; \quad \overline{Z}_{\alpha,j} = \prod_{i \in \overline{C}_{\alpha,j}} z_i,$$

(17)

along the red and blue shrunk lattices.

The logical operators for the canonical color code is made up of a single type of Pauli operator as we have just seen. We will see that in some of the non-CSS versions this property changes with the presence of logical operators that are composed of more than one type of Pauli operator. These models can be distinguished using the idea of $P$-distances that is defined as follows. In the stabilizer formalism, the code distance is taken to be the minimum weight of a symmetry operator that cannot be expressed as a product of the stabilizers that make up the commuting projector Hamiltonian. If this minimum weight symmetry operator is composed of a single type of Pauli operator, either $x$, $y$ or $z$, then this is called as a $P$-type logical and the corresponding weight is denoted the $P$-distance.

### 3. Elementary excitations

Consider single-qubit bit-flip or phase-flip operations on a vertex of the trivalent lattice. They excite either three or six stabilizers as opposed to the toric code where the same excites two or four stabilizers. More importantly, it is not possible to excite an odd number of stabilisers for the toric code defined on a closed manifold. Before studying the nature of the ‘triad excitations’ we list them for a unit cell of the lattice in Fig. 5 in the form of a Table I. Further, the triad excitations are immobile, i.e., they cannot be moved around the lattice using string operators like the deconfined anyons of the abelian toric code.

### 4. Deconfined anyons

The topological character of the model warrants the existence of anyonic excitations just as in the case of the abelian toric code. They are obtained from a string of $x$ or $z$ operators acting in the red and blue shrunk lattices. The latter creates the ‘electric charges’ and the former creates the ‘magnetic fluxes’ of the theory. We have

$$e, \quad E_{\gamma} = \prod_{j \in \gamma} x_j; \quad e, \quad E_{\overline{\gamma}} = \prod_{j \in \overline{\gamma}} x_j$$

$$m, \quad \overline{M}_{\gamma} = \prod_{j \in \gamma} z_j; \quad m, \quad \overline{M}_{\overline{\gamma}} = \prod_{j \in \overline{\gamma}} z_j,$$

(18)

where $\gamma$ and $\overline{\gamma}$ are paths in the red and blue shrunk lattices respectively as shown in Fig. 3. As seen earlier the string operators on the green shrunk lattice can be obtained as a product of the string operators on the red and blue shrunk lattices and hence $e$ and $m$ are not independent excitations. The anyons can also be obtained by combining elementary excitations (See Table I) on vertices of a shrunk lattice of a chosen color. Thus the anyonic content of the color code is isomorphic to that of the $\mathbb{Z}_2 \times \mathbb{Z}_2$ toric code as seen from Eq. (18). However, the spectrum of the color code also con-
tains the immobile ‘triad’ excitations which are not found in the $\mathbb{Z}_2 \times \mathbb{Z}_2$ toric code. While the anyons of the theory account for the long range properties of the system, the immobile excitations represent the short range physics. In particular they have no effect on the topological parts of the GSD and the entanglement entropy.

We will see that the energetics of single qubit excitations around a trivalent vertex can be altered resulting in new stabilizer codes that are non-CSS by construction. Thus the elementary excitations can be thought of as the starting point of the non-CSS color codes.

C. Unitaries permuting the Pauli operators

To check the equivalences of the different non-CSS color codes we require the unitary operators that permute the Pauli matrices, $\{ x, y, z \}$, and there are precisely 6 of them. We systematically construct these operators in terms of the Hadamard, $H$ and phase gates, $S$.

1. The Hadamard operator, $H = \frac{1}{\sqrt{2}} [ x + z ]$, satisfies, $H^2 = 1$, and has the following action on the Pauli operators under conjugation,

$$
\begin{align*}
\{ x \to z, \\
y \to -y, \\
z \to x
\}
\end{align*}
$$

(19)

2. The phase gate, $S = \left( \begin{array}{cc} 1 & 0 \\ 0 & i \end{array} \right)$, satisfies $S^4 = 1$, and permutes the Pauli matrices as,

$$
\begin{align*}
\{ x \to y, \\
y \to -x, \\
z \to z
\}
\end{align*}
$$

(20)

3. The operator, $t = \frac{1}{\sqrt{2}} [ y + z ] = SHS^3$, acts like the ‘Hadamard’ gate interchanging, $y$ and $z$,

$$
\begin{align*}
\{ x \to -x, \\
y \to z, \\
z \to y
\}
\end{align*}
$$

(21)

4. The unitaries, $C_1 = H S^3$ and $C_2 = SH$, cyclically permute the Pauli matrices,

$$
\begin{align*}
C_1 : \{ x \to y, \\
y \to z, \\
z \to x
\}
\end{align*}
$$

(22)

and

$$
\begin{align*}
C_2 : \{ x \to z, \\
y \to x, \\
z \to y
\}
\end{align*}
$$

(23)

5. Finally we have the identity transformation which leaves all the Pauli matrices unchanged.

Note the appearance of the ‘$-1$’s in the above transformations. It is worth observing that the above unitaries do not form the permutation group of three objects, $S_3$. These are not the only operators that permute the Pauli operators. For example using the transpositions, $\tilde{H} = \frac{1}{\sqrt{2}} (z - x)$ and $\tilde{t} = \frac{1}{\sqrt{2}} (y - z)$ interchange $x \leftrightarrow z$, and $y \leftrightarrow z$ respectively, up to a sign. These operators can be expressed in terms of the Hadamard and phase gates as, $\tilde{H} = S^2 HS^2$ and $\tilde{t} = -S^3 HS$ respectively.

III. OVERVIEW OF THE MAIN RESULTS

This section serves a dual purpose of summarizing our findings and also as a guide to navigate the paper.

- The stabilizer codes to follow are built from the ground up by considering the placement of Pauli operators, $x, y$ and $z$ around vertices and along edges of a face appearing in the trivalent lattice. For the vertex configurations we are guided by the local energetics of the Hamiltonian formed out of the stabilizers of the code (dubbed elementary excitations earlier). By considering vertices with unequal energy configurations we are guaranteed inequivalent stabilizer codes that cannot be mapped to each other via LU’s. As a result we find two more color code-like stabilizer codes on trivalent lattices that share the same topological properties as the latter. We label them as the $[444]$- and $[345]$-color codes based on the possible energy excitations around the trivalent vertices of the code. From this perspective the original color code takes the notation, $[336]$-color codes. The $P$-distance of the code sets the requirement for the configurations along the edges. We obtain three types of edge configurations, homogeneous ($P$-distance is zero), partially mixed ($P$-distance of some of the string logical operators are 0 and some $\infty$) and finally fully mixed ($P$-distance for the string logical operators is $\infty$). The vertex and edge configurations form the building blocks of the code (See Sec. IV).

- As examples the full non-CSS stabilizer codes are built on the hexagon ($6,6,6$) and the square-octagon ($4,8,8$) lattices. We show the construction of the $[444]$-color codes on both these lattices and with three different edge configurations, homogeneous, partially and fully mixed. Moreover we provide the codes with and without translational invariance. These can be found in Sec. V.

- In the case of the $[345]$-color codes, the combinations of vertex and edge configurations result in two types of models: deconfined and confined models. In the latter elementary string operators of a given color excite either two or three stabilizers of that color leading to ‘energetic’ strings, that is if the string is extended along certain directions all the
stabilizers along the string get excited apart from the end points. Whereas the former case is the more familiar one occurring in the surface codes and the original color code where the strings creating anyons excite only the stabilizers associated to the faces at the endpoints of the string resulting in its deconfined nature. In other words there is no cost for moving anyons in the deconfined [345]-color codes and there is an energy cost for moving anyons in the confined counterpart. Furthermore there are three inequivalent [345] vertices that cannot be mapped to each other using a LU (See Fig. 12). An example of each of these different types of [345]-color codes on the hexagon and the square-octagon lattice for the three types of edge configurations, with and without translational invariance, can be found in Sec. V.

IV. GENERAL VERTEX AND EDGE CONFIGURATIONS FOR TRIVALENT LATTICES

Our goal is to provide a detailed construction of non-CSS variants of the color codes on both a finite trivalent lattice tiling a closed two-dimensional surface and on finite trivalent lattices with boundaries. Every such lattice consists of vertices $V$ and edges $E$ that make up the faces $F$ and on each face we consider two stabilizers, just as in the color code. Henceforth, we refer to them as the stabilizer pair. As in the color code each face on the trivalent lattice contains an even number of vertices and so we only use squares, hexagons, octagons and so on to tile the surface. The common regular lattices tiling the plane include

- the hexagon (6, 6, 6) lattice
- the square-octagon (4, 8, 8) lattice
- the square-hexagon-dodecagon (4, 6, 12) lattice.

We begin the construction of a non-CSS model by considering vertex and edge configurations made out of the three Pauli operators. These are then joined together resulting in stabilizer pairs for each face forming an abelian subgroup of the Pauli group. This construction results in a commuting stabilizer Hamiltonian. Before diving into the configurations we look at the types of vertices possible from the trivalent lattice geometry and the types of edge configurations possible from the desired physical properties of the code.

Vertices come in three kinds, bulk, boundary and corner. While lattices on closed surfaces consist of only trivalent bulk vertices, finite lattices consist of trivalent vertices in the bulk, vertices that end on the boundaries of the lattice and those that are located at the corners of the lattice (See Fig. 6). The bulk and boundary type vertices are trivalent whereas the corner type vertices are bivalent. Corner vertices share a single face, whereas boundary and bulk vertices share two and three faces respectively.

**Figure 6.** The three types of vertices on a finite trivalent lattice. Only a portion of the entire lattice is shown.

When building edge configurations we keep in mind that the resulting stabilizers have to generate an abelian subgroup of the Pauli group. This is very restrictive, reducing the number of possibilities. Furthermore as the logical operators of the model are built using strings of Pauli operators along edges, we distinguish edge operators according to whether or not they can give rise to P-type operators. This gives us three cases, fully mixed, partially mixed and homogeneous (no mixing) edge configurations. By making these choices for the edge configurations we can either end up in a model where

- the P-distance for the string logicals is infinity when the edge configurations are fully mixed,
- a finite distance P-type operator for some of the string logicals when the logicals are partially mixed,
- all logicals are of finite P-distance when the logicals are homogeneous or pure.

Finally, as a guiding principle that will help classify the space of non-CSS color codes, we impose another condition on the choice of the Pauli operators on a particular qubit on a given stabilizer pair which we call the anticommuting rule - The Pauli operators for each qubit for a given stabilizer pair must anticommute.

It is easy to verify that the stabilizer pairs of the canonical color code satisfy this condition. Imposing this rule picks out the models which share the topological properties of the canonical color code. The reason for this is tied to the quantum double structure of the algebra of operators appearing in the toric code and the color code [5, 6]. By requiring this condition we preserve this structure for the stabilizers of the non-CSS variants.

We now turn to systematically exhausting the possible configurations for the three types of vertices and the edges. These can be considered as the building blocks for constructing the full non-CSS color code variants.
A. Bulk vertex configurations

A crucial feature of the color codes is the single qubit Pauli operators that result in its elementary excitations (See Table I). In particular, for the canonical color codes, we see that the $x$, $y$ and $z$ operations violate exactly 3, 6 and 3 stabilizers shared by the trivalent vertex respectively. The fact that we can excite an odd number of stabilizers by local operations distinguishes the color codes from surface codes, such as the toric code. Thinking of the (363) as a local energy configuration of the color code we are led to the possibility of finding other energy configurations that can arise from an alternative choice of stabilizers.

To begin a systematic search for the different models we consider a trivalent bulk vertex (See Fig.7) sharing three faces or six stabilizers, two for each face. As there are 6 positions to fill with the Pauli $x$, $y$ and $z$ operators we consider the integer solutions of

$$a + b + c = 6,$$

where $a$, $b$, $c$ are the number of $x$, $y$ and $z$ operators appearing around the vertex. The possible configurations are shown in Table II.

While the models arising from the possible bulk vertex configurations in Table II can exhibit different features, we are interested in those possessing topological properties of the canonical color code. These are precisely the configurations that obey the *anticommuting rule*. This condition drastically reduces the number of possibilities for the number of configurations that we have to consider.

It is easy to see that only the configurations in rows 5 to 7 of Table II can possibly satisfy this requirement and so we will only study the models arising from them in the main text. Giving up this condition results in additional lower weight symmetries, for example in the configurations of row 4 of Table II. This results in models with an extensive ground state degeneracy or an enlarged dimension for the code space. We ignore these possibilities in the main text, relegating some of their features to App. G.

Table II. Possible configurations at each trivalent vertex. The first column shows the partitions of six which indicate the possible number of the different types of Pauli operators, and the second column indicates the corresponding configuration. The third column specifies the number of stabilizers violated around each vertex upon the application of the Pauli operators, $x$, $y$ and $z$ respectively. The energy configurations are seen as a partition of 12. There is no LU mapping a configuration on some row to a configuration belonging to a different row as the corresponding Hamiltonians have different spectra. The canonical color code and its variants connected by LU’s are shown in row 5.

We denote the models belonging to row 5 of Table II as the [336]-color codes. The bulk vertices in this set will be denoted by [336]-vertices. Each of these vertices could have configurations from one of the three options shown in row 5. In a similar manner the models from row 6 of Table II are grouped into the set [345]-color codes and the corresponding bulk vertices are denoted as the [345]-vertices. Finally the models and their bulk vertices in row 7 of Table II are denoted the [444]-color codes. The bulk vertices in this set will be denoted by [444]-vertices, respectively.

In the following subsections, we will systematically go through operator configurations for the different bulk vertex types mentioned in Table II.

1. [444]-vertices

Each [444] bulk vertex consists of two Pauli, $x$, $y$ and $z$ operators as shown in Table II. There are two types of bulk vertex configurations:

- **Type I** - There are $x$, $y$ and $z$ around every vertex for the stabilizers containing that vertex
- **Type II** - Two of the Pauli operators are the same and differ from the third on each of the associated stabilizers containing that vertex.
An example of each of these types is illustrated in Fig. 8. Our task now is to find the different configurations corresponding to these two possibilities and group them into different inequivalent classes, that is those that are not related to each other by an LU.

**Type I** - We begin with the cyclic configurations around the bulk vertex for the two sets of stabilizers. There are precisely six sets of such cyclic configurations corresponding to the six permutations of \(x, y\) and \(z\). These are \(\{(xyz), (xzy), (yxz), (yzx), (zxy), (zyx)\}\), read clockwise around the bulk vertex beginning from position 1 (See Fig. 7). Using the requirement of anticommuting operators we obtain two sets for the pairs of bulk vertex operators, 

\[
A = \left\{ \left( \begin{array}{c} xyz \\ zyx \end{array} \right) \right\}, \text{ } \text{ } B = \left\{ \left( \begin{array}{c} xzy \\ yzx \end{array} \right) \right\}. \tag{24}
\]

Before we look for the inequivalent pairs among these three sets we notice that each of these sets can be mapped into each other by a permutation of the \(x, y\) and \(z\) operators and hence it is enough to consider any one of these sets. Take for example a set in Eq. (25) mapping to the

\[
\left\{ \left( \begin{array}{c} xyz \\ zyx \end{array} \right) \right\}.
\]
set in Eq. (26), by interchanging \( y \) and \( z \), and to the set in Eq. (27), by interchanging \( x \) and \( y \). Hence we look for the inequivalent pairs of stabilizers in the set in Eq. (25) and we find three of them,

\[
\left\{ \begin{array}{ccc}
(xxy & yzz & xxy) \\
(yxz & yzy & yzy)
\end{array} \right\}.
\] (28)

Along with the lone cyclic configuration in Eq. (24), we obtain in total four classes of inequivalent vertex configurations as shown in Fig. 10.

**Shuffling stabilizers -** Each bulk vertex shares three stabilizers corresponding to the three colors adjacent to this vertex. We can interchange the stabilizers of the same color and this amounts to an interchange of the Pauli operators in the bulk vertex configuration for the pair. This can also be thought of as relabelling the indices I and II. This operation leaves the Hamiltonians built out of these stabilizers invariant as it translates to a shuffling of stabilizers in the Hamiltonian. Taking this into account, we see that each of the four classes of the [444] bulk vertices can be mapped into each other as shown in Fig. 11. Repeating this for the other classes we conclude that there is just a single type of bulk vertex in the [444] case. Any other type of [444] bulk vertex can be mapped into this one by a LU.

**2. [345]-vertices**

Now we find the inequivalent pairs of [345] bulk vertices by looking at the configurations in row 6 of Table II. Among the six possible configurations in row 6, namely, \{xyyzzz, xyyzzz, xxyzzz, xxyyyz, xxyyz, xxyyz\}, we need to consider just one of them as the rest can be obtained by permutations of the Pauli operators and are thus LU equivalent to each other. For example \( xxyyz \) can be obtained from \( xyyzzz \) by replacing \( x, y, z \) with \( z, x, y \), respectively. Thus we consider permutations of \( xyyzzz \) to build the stabilizers around the [345] bulk vertices.

![Figure 10. The four inequivalent [444] bulk vertex pair operator configurations from Eqs. (24) and (28).](image)

**Figure 11. The equivalences between the four [444] bulk vertex configurations.**

**We have three sets of operator pairs that satisfy the anticommuting rule for the \( xyyzzz \) configurations,**

\[
A = \left\{ \begin{array}{ccc}
(xyy) & (xyx) & (xxy) \\
(zzz) & (yzy) & (zyz)
\end{array} \right\}
\] (29)

\[
B = \left\{ \begin{array}{ccc}
(yxy) & (zxy) & (yxz) \\
(zzz) & (yzy) & (zyz)
\end{array} \right\}
\] (30)

\[
C = \left\{ \begin{array}{ccc}
(yyx) & (yxx) & (xyx) \\
(zzz) & (yzy) & (zyy)
\end{array} \right\}
\] (31)

Notice that in each set given by Eqs. (29), (30), (31) the operator pairs can be obtained from each other by shuffling. Thus each set can be represented by a single pair of operators. On the other hand to relate the operator pairs belonging to different sets we need to rotate the bulk vertex. This feature is different from what we saw for the [444]-vertices and it leads to three inequivalent models on an arbitrary trivalent lattice. The three inequivalent [345] bulk vertices are shown in Fig. 12.

3. [336]-vertices

Bulk vertices corresponding to the canonical color code belong to the [336] type and are listed in row 5 of Table II. Analogous to the arguments in the earlier cases it is easy to see that it is enough to consider the pairs of stabilizers arising from considering bulk vertices that include \( xxxxzzz \) as other configurations are obtained by permuting the Pauli matrices. The possible configurations obtained by rearranging the three \( x \) and \( z \) operators around the bulk vertex obeying the anticommuting rule are

\[
\left\{ \begin{array}{ccc}
(xxx) & (xxx) & (xxx) \\
(zzz) & (zzz) & (zzz)
\end{array} \right\}
\] (32)

However, it is rather easy to check that there is just one inequivalent class in this set as the four pairs of
operators can be mapped into each other by a simple rearrangement of the stabilizers. Thus we conclude that there is just one [336] bulk vertex possible and every other configuration is LU equivalent to this one.

There is one equivalence class of bulk vertex operator configuration for the [444] and [336] vertex types and three inequivalent ones for the [345] bulk vertex type.

B. Boundary vertex configurations

For the vertices ending on the boundaries of the finite lattice (See Fig. 13), there are just four places to fill with the Pauli operators as opposed to the six spots around the bulk trivalent vertices. Thus possible configurations of the Pauli operators in this case are the integer solutions of

\[ a + b + c = 4. \]

However, if we take into the account the anticommuting rule we are left with just two sets of solutions, namely, the rows 3 and 4 in Table III. While there are three possibilities for the operators in each case, they can be mapped into each other using LU’s. So we take the \{(2,1,1),(2,2,0)\} configurations as the representative elements in this class. Upon the application of Pauli operators on the boundary vertex qubits, leads to an energy configuration of either \{[233],[224]\} for the \{(2,1,1),(2,2,0)\} boundary vertices respectively as seen from Table III). We label these boundary vertices by its energetics just as in the case of the bulk vertex configurations. We explore further details of these two vertex types in the following subsections.

1. [233]-vertices

There are three choices for the vertices, namely \(xyzz, xyyz\) and \(xxyz\) which are LU equivalent to each other. Considering \(xyzz\), we find two sets of stabilizer pairs that are compatible with the anticommuting rule,

\[
A = \left\{ \begin{array}{c}
  \begin{bmatrix}
      xy \\
      zz
    \end{bmatrix} \\
    \begin{bmatrix}
      yz \\
      zz
    \end{bmatrix}
\end{array} \right\} \quad (33)
\]

\[
B = \left\{ \begin{array}{c}
  \begin{bmatrix}
      xz \\
      yz
    \end{bmatrix} \\
    \begin{bmatrix}
      zz \\
      yz
    \end{bmatrix}
\end{array} \right\}. \quad (34)
\]

The operator pairs in Eq. (33) are mapped into each other with a LU and so are the operator pairs in Eq. (34). However, there is no LU between these two sets imply-
ing there are two inequivalent classes of [233] boundary vertices as shown in Fig. 14. Nevertheless, they can be transformed into each other by a shuffling of the stabilizers.

![Figure 14. Boundary vertex configurations of the [233] type. They are inequivalent up to an interchange of the stabilizers.](image)

2. [224]-vertices

We have three choices for the configurations, \( xxyy \), \( yyzz \), \( zxxx \) that are LU equivalent to each other. There are two sets of stabilizer pairs resulting from \( xxyy \),

\[
\left\{ \begin{pmatrix} xx \\ yy \end{pmatrix}, \begin{pmatrix} xy \\ yx \end{pmatrix} \right\}. \tag{35}
\]

These are mapped into each other by a shuffling of the stabilizers as in the earlier case. These configurations are shown in Fig. 15.

![Figure 15. Boundary vertex configurations of the [224] type. They are inequivalent up to an interchange of the stabilizers.](image)

C. Corner vertex configurations

At the corners of a finite lattice we have just two positions for the operator pair to be filled with the Pauli operators. With the anticommuting property we find that there are precisely three possibilities for the operator pairs at the corner, namely, \( x \) and \( y \), or \( y \) and \( z \), or \( z \) and \( x \) (See Fig. 16).

![Figure 16. The three possible corner vertex configurations obeying the anticommuting rule. Note that the corner vertices are bivalent.](image)

D. Edge configurations

As mentioned earlier there are three types of edge configurations that could lead to models with varying P-distances for the logical operators. For each edge on a stabilizer pair (See Fig. 17) we have four possible positions, with each position containing two neighbouring qubits. Thus for each position we require two Pauli operators and we choose them such that they satisfy the anticommuting rule and they lead to commuting stabilizers.

The fully mixed logicals that result in models with infinite P-distance are obtained from the edge configurations,

\[ \{(xy), (yz), (zx)\} \] or \[ \{(yx), (zy), (xz)\}. \tag{36} \]
The four positions for an edge in the stabilizer pair can be filled using the configurations in either of the sets. The configurations giving partially mixed logicals are either \( \{(xz), (yz), (zy)\} \) or \( \{(xz), (yy), (zx)\} \) or \( \{(xy), (yx), (zz)\} \).

Finally the choice leading to no mixing or an homogeneous edge configuration is obtained using
\[
\{(xx), (yy), (zz)\}. \tag{38}
\]

An example for each case is illustrated in Fig. 18.

![Figure 18. An example for each type of edge configuration leading to logical operators with different P-distances. Note that the stabilizers formed out of these edge configurations will commute with each other. (a) Homogeneous configuration where there is no mixing of the Pauli operators. (b) Partially mixed edge configuration. (c) Fully mixed edge configuration.](image)

There are three types of edges depending on the mixing of the four operator positions with respect to the stabilizer content, namely, homogeneous, partially mixed, and fully mixed.

V. NON-CSS COLOR CODES ON HEXAGONAL LATTICES

We are now in a position to build the full stabilizer codes by combining the inequivalent vertex and the different edge configurations according to what type of code we require. The construction works for both regular and irregular trivalent lattices, i.e., with and without translational invariance. We can work on both finite lattices with boundaries and on lattices that discretize a closed surface. We begin with the hexagonal lattice on a torus.

The construction of the stabilizer pairs for the hexagonal lattice proceeds by consistently joining the vertex and edge configurations keeping in mind the requirements of the code and this is best illustrated through figures. We will carry out the procedure for each vertex type separately in what follows. We will first show the method locally in each case and then give examples on a full lattice. By local construction we mean that the construction is carried out on a small part of the lattice, mostly on two or three edges that do not form a closed figure. While it is always possible to do this for the different vertex types and edge configurations the method fails in some instances when we try to close the figures. In this section we will only provide the definitions of the various non-CSS color codes and leave the analysis of their properties in sections following this.

A. Construction of [444]-color codes

The local construction proceeds using the following steps:

1. Start with a pair of two adjacent trivalent [444]-vertices connected by an edge and choose a [444]-vertex for one of the trivalent vertices. This choice is arbitrary and there is no loss of generality for a different choice.

2. Fill the edge with an allowed edge configuration. This can be done in three ways; by placing either a fully mixed edge configuration (there are two choices for this) or a partially mixed edge configuration (there are three choices for this) or an homogeneous edge configuration.

3. Complete the configuration on the second vertex such that it is also of the [444]-vertex type.

4. Using the second [444]-vertex we can continue the above steps to further extend the code on the lattice.

The procedure for the construction is summarized in Fig. 19.

While this procedure can be continued in a consistent way for an arbitrary trivalent lattice with or without boundaries, we will look at examples of trivalent lattices such as the (6, 6, 6)- and (4, 8, 8)-lattices.

a. Hexagonal lattice - We can construct consistent [444]-color codes both with and without translational invariance on the hexagon lattice. Starting with the former, the stabilizers for the simplest [444]-color code that we can write down on the hexagon lattice are shown in Fig. 20. This model contains edge configurations of the homogeneous type and around every vertex we have all the three Pauli operators. We will denote
Figure 19. The different steps involved in the construction of the [444]-color code in a local part of a trivalent lattice. In step 2 (a) homogeneous, (b) partially mixed and (c) fully mixed edge configurations are shown. There are more possibilities at step 4 and we show just one of them.
this model as the canonical $[444]$-color code. Every red face has the same set of stabilizers as the original color code, whereas on the blue (green) faces we have $Y$ and $Z$ ($X$ and $Y$) type of stabilizers. It is clear that these stabilizers can also be defined on an arbitrary trivalent, tricolorable lattice.

![Figure 20](image)

Figure 20. The canonical $[444]$-color code. There is no mixing in each edge for this case.

We can now construct the $[444]$-color codes where the edge configurations are partially or fully mixed as shown in Fig. 21. The models with partially and fully mixed edge configurations are equivalent to the canonical $[444]$-color code by an LU. This is demonstrated explicitly in Fig. 22.

![Figure 21](image)

Figure 21. The $[444]$-color code with edges that are partially mixed (a) and fully mixed (b).

The non-translationally invariant $[444]$-color codes with partially and fully mixed edge configurations are shown in App. B1. The codes on the square-octagon lattice, with and without translational invariance, can be found in App. E.

**B. Construction of $[345]$-color codes**

The construction here goes along the lines of the $[444]$-color codes. However there are two crucial differences from the $[444]$ case. First, there are three types of inequivalent $[345]$ bulk vertices to choose from, Eqs. (29), (30) and (31). We will begin our construction from the first configuration in each of these inequivalent classes guaranteeing three inequivalent stabilizer codes. Second, we have more possibilities for combining the chosen bulk vertex with an edge configuration. We find two types of combined vertex-edge configurations.

1. The combined vertex-edge configurations admit deconfined anyons as in the color codes. That is either two or four stabilizers are excited by a local string configuration exciting the end points of the string as shown in Fig. 23. String operators formed out of the edge configurations in each of the figures in Fig. 23 lead to deconfined anyons. For example in Class I (b), the string operators, $(yx)$, $(zz)$ excite two of the four stabilizers at the end points.

![Figure 22](image)

Figure 22. (a) The LU between the translationally invariant $[444]$-color code with (a) partially mixed, (b) fully mixed edge configurations and the canonical $[444]$-color code.

![Figure 23](image)

Figure 23. The deconfined edge configurations possible for three inequivalent $[345]$-color codes (See Fig. 12), indicated by an orange dot. The edge configurations can be either (a) homogeneous, (b) partially or (c) fully mixed.
of the string, and the operator \((xy)\) excites all the four stabilizers. This is similar to what occurs in the \([336]\) color code and the \([444]\)-color codes introduced earlier. We will denote the models built out of these as the *deconfined* \([345]\)-color codes.

2. The vertex-edge combinations lead to confined anyons along the direction defined by the edge. This follows when the local string configurations excite either two or three of the four stabilizers located at the end points of the string (See Fig. 24). String operators formed out of the edge configurations in each of the figures can lead to confined anyons. For example in Class I (b), the string operators, \((yx)\), \((xy)\) excite three of the four stabilizers at the end points of the string, and the operator \((zz)\) excites two of the four stabilizers. Extending this string along a particular direction can lead to an *energetic string* that not only excites the stabilizers at the end points of the string but also the stabilizers along the path of the string. Depending on the rest of the code it is possible to create deconfined and confined anyons with appropriate choices for the string operators. We will call the models built out of these configurations as the *confined* \([345]\)-color codes.

a. **Deconfined** \([345]\)-color codes - The steps for the local construction in the deconfined setting are illustrated in Fig. 81. As in the \([444]\)-color codes we can construct stabilizer codes with or without translational invariance. The construction also holds for an arbitrary trivalent, tricolorable lattice as it is local. We will call the codes where none of the edge configurations are mixed as the *canonical* \([345]\)-deconfined color codes.

We can have three inequivalent translational invariant codes on the hexagon lattice corresponding to the \([345]\)-bulk vertices shown in Fig. 12. The edge configurations are such that the vertex-edge combines are of the deconfined type as shown in Fig. 23. Furthermore the edge configurations can be either homogeneous, partially or fully mixed as shown in Fig. 25.

b. **Confined** \([345]\)-color codes - Fig. 82 shows the steps for the local construction of the confined \([345]\)-color codes. Using this we can proceed to construct the confined \([345]\)-color codes on different lattices. However, we do not obtain consistent color codes for any even sided face in the lattice. By this we mean that either not all the vertex-edge combines are confined, i.e. some come out to be deconfined, or not all the vertices are of the \([345]\) bulk vertex type. For example while it is possible to define a
confined \([345]\)-color code on the square and hexagon it is not possible to obtain an octagon or a decagon (See App. B3).

The consistent models on the hexagonal lattice, with translational invariance for Class I, is shown in Figs. 27, models in the classes II and III with translational invariance are explained in App. D and models without translational invariance are shown in Figs. 78 and 79 of App. B3.

C. Construction of \([336]\)-color codes

For completion we show how this construction procedure works for the \([336]\)-color codes that are equivalent to the original color code by an LU. As a result we find color codes that have mixed edge configurations. We begin with the local construction of this type in Fig. 83 in App. C.

Using this the non-CSS \([336]\)-color codes on the hexagonal lattice for partially and fully mixed edge configurations, with translational invariance is shown in Figs. 28 and the non-translationally invariant analogue is shown in Fig. 80 of App. B4.

VI. \(\mathbb{Z}_2 \times \mathbb{Z}_2\) TOPOLOGICAL ORDER OF NON-CSS COLOR CODES

So far we have studied the general construction of stabilizer codes on trivalent lattices and explored how to apply this construction to generate different types of color codes in two-dimensional lattices. An important feature we used was the bulk vertex energetics to make a broad classification of the non-CSS color codes. In other words, we exploited the local ‘short-range’ physics to obtain various models.

In this Section, we study the long-range properties of the canonical \([444]\) and \([345]\)-color codes on the hexagonal lattice and show that they realize the \(\mathbb{Z}_2 \times \mathbb{Z}_2\) topological phase just as the \([336]\)-color codes that contain,

\begin{align*}
\text{Set I :} & \{r_I, b_I, g_I\} \\
\text{Set II :} & \{r_{II}, b_{II}, g_{II}\}.
\end{align*}

While this notation is useful to describe the stabilizers on each of the colored faces in the non-CSS color codes with partially mixed and fully mixed edge configurations, it is not needed for the codes with homogeneous edge configurations. In the latter case we will denote the stabilizers as either of the Pauli operators, \(X\), \(Y\) or \(Z\) with the appropriate color.

The Hamiltonian, is given by

\begin{equation}
H = -\sum_{j=1}^{f} (Pr_{Ij} + Pr_{IIj}) - \sum_{j=1}^{f} (Pb_{Ij} + Pb_{IIj}) \\
- \sum_{j=1}^{f} (Pg_{Ij} + Pg_{IIj}),
\end{equation}

with \(j\) acting as the index for the faces and \(f\), \(f\) and \(f\) denoting the number of red, blue and green faces re-
respectively. Note that for a general trivalent, tricolorable lattice $f \neq f \neq f$. They are equal in the case of a hexagonal lattice.

The GSD can be computed either by evaluating the trace of the projector to the ground state manifold, or by identifying the constraints satisfied by the stabilizers to obtain the number of independent stabilizers. We will find that there are precisely $2|F| - 4$ independent stabilizers for both the [444]-color code and the deconfined [345]-color code. This implies that the number of encoded qubits is the same as the original CSS color code, $|V| - 2|F| + 4 = 4g$, with $g$ being the genus. We will also write down the logical Pauli’s in each case and account for the GSD. Along this process we also provide a description of the anyon content for the different codes that indicates the type of topological phase.

We primarily work on the hexagonal lattice and for the case of the translationally invariant stabilizer codes. We believe the topological properties continue to hold for the non-translationally versions of these codes as well.

### A. The [444]-color codes

The stabilizers of the canonical [444]-color codes is shown in Fig. 20. As all the edge configurations are homogeneous, we can think of the code simply as green faces made up of the $X$ and $Y$ stabilizers, the blue faces with $Y$ and $Z$ stabilizers and the red with $Z$ and $X$ stabilizers. This makes contrast to the original color code where the $Z$ and $X$ stabilizers make up all three colors.

#### 1. Ground State Degeneracy

The constraints among the stabilizers of this model are easily read off from the code in Fig. 20 as,

\[
\prod_{j=1}^f r_l \prod_{j=1}^f b_{ll} = \prod_{j=1}^f r_l \prod_{j=1}^f g_l \prod_{j=1}^f g_{ll} = 1, \tag{42}\]

\[
\prod_{j=1}^f b_{ll} \prod_{j=1}^f g_{ll} = \prod_{j=1}^f b_l \prod_{j=1}^f r_l \prod_{j=1}^f r_{ll} = 1, \tag{43}\]

\[
\prod_{j=1}^f g_l \prod_{j=1}^f r_{ll} = \prod_{j=1}^f g_l \prod_{j=1}^f b_l \prod_{j=1}^f b_{ll} = 1, \tag{44}\]

\[
\prod_{j=1}^f r_{ll} \prod_{j=1}^f b_l \prod_{j=1}^f f = \prod_{j=1}^f b_{ll} \prod_{j=1}^f r_l \prod_{j=1}^f r_{ll} = 1, \tag{45}\]

\[
\prod_{j=1}^f b_{ll} \prod_{j=1}^f g_l \prod_{j=1}^f g_{ll} = 1, \tag{46}\]

\[
\prod_{j=1}^f g_{ll} \prod_{j=1}^f r_l \prod_{j=1}^f r_{ll} = 1, \tag{47}\]

Out of these eleven constraints we can verify that only four are independent just as in the case of the original [336]-color code. We make a choice to pick the following operators

\[
\beta_1 = \prod_{j=1}^f r_l \prod_{j=1}^f b_{ll}, \quad \beta_2 = \prod_{j=1}^f b_l \prod_{j=1}^f g_{ll}, \tag{49}\]

with the constraints given by

\[
\beta_i = 1, \quad \forall i = 1 \text{ to } 4. \tag{50}\]

We observe that the second constraints of Eqs. (43) and (44) are obtained as $\beta_2 \beta_3 \beta_4 = 1$ and $\beta_1 \beta_2 \beta_4 = 1$ respectively. Similarly, the constraints in Eqs. (45)-(47) are equivalent to $\beta_1 \beta_2 \beta_3 = 1$, $\beta_1 \beta_4 = 1$ and $\beta_3 \beta_4 = 1$, respectively. The two constraints in Eq. (48) are $\beta_2 \beta_4 = 1$ and $\beta_1 \beta_3 \beta_4 = 1$ respectively.

So the number of encoded qubits is $|V| - 2|F| + 4$ which is 4 for the hexagonal lattice tiling on a torus. On more general trivalent lattices tiling of a genus-$g$ surface, $|V| - 2|F| + 4$ reduces to $4g$ as seen earlier. Thus, we conclude that the GSD of the [444]-color codes coincides with the original [336]-color code.
2. The Code Space

Now we construct the ground states or the encoded qubits for [444]-color code with homogeneous edge configurations on the hexagonal lattice. The construction illustrated here goes through for a more general trivalent, tricolorable lattice discretizing surfaces of arbitrary genus with minor modifications. We work with the stabilizers, $X$, $Y$ for the green faces, $Y$, $Z$ for the blue faces and $Z$, $X$ for the red faces.

As in the construction of the encoded qubits for the original CSS $[336]$-color code, we project a state diagonalizing one set of the stabilizers (the seed state), say the $Z$ stabilizers, with the remaining stabilizers onto the code space. This procedure applies for the [444]-color code as well albeit a crucial difference coming from the choice of the state to be projected. It can be checked that choosing a homogeneous configuration for the seed state, namely a state containing either $|0_z\rangle$, $|0_x\rangle$ or $|0_y\rangle$ on all the vertices is annihilated by the projectors built out of the stabilizers supported on the hexagonal faces. For example $PX_PY |s_Z\rangle$, with $|s_Z\rangle$ being the state with $|0_z\rangle$ on the vertices of the hexagon, is 0. Notice that this is non-zero for a square, octagonal or any face whose number of vertices is a multiple of four.

As a result we choose a seed state such that there is one $|1_z\rangle$ state on every green face, with the remaining vertices filled by $|0_z\rangle$. This is attained by a seed state, $|s_{[444]}\rangle$ shown in Fig. 29. This choice ensures that some LU’s. In this regard we notice that the seed state in Fig. 29, $|s_{[444]}\rangle$ is obtained from $|s_Z\rangle$, the seed state of the CSS $[336]$-color code by flipping the $|0_z\rangle$’s to $|1_z\rangle$’s along alternating pairs of green faces (shown as green edges in Fig. 29). We denote these edges as $e$ in the formulas below. That is,

$$|s_{[444]}\rangle = \prod_{v_i,v_j \in e} x_{v_i}x_{v_j} |s_Z\rangle.$$  

(51)

The encoded qubit is obtained from

$$|0_1,0_2,0_3,0_4\rangle_{[444]} = \prod_{j=1}^f PY_j \prod_{j=1}^f PX_j$$

$$\prod_{j=1}^f PX_j PY_j |s_{[444]}\rangle,$$

$$= \prod_{v_i,v_j \in e} x_{v_i}x_{v_j} \prod_{j=1}^f PY_j \prod_{j=1}^f PX_j$$

$$\prod_{j=1}^f PX_j PY_j |s_{Z}\rangle,$$

$$= \prod_{v_i,v_j \in e} x_{v_i}x_{v_j} \prod_{j=1}^f PX_j \prod_{j=1}^f PX_j$$

$$\prod_{j=1}^f PX_j |s_{Z}\rangle,$$

$$= U |0_1,0_2,0_3,0_4\rangle_{[336]},$$

(52)

where

$$U = \left( \prod_{v_i,v_j \in e} z_{v_i}z_{v_j} \right) \left( \prod_{v_i,v_j \in e} x_{v_i}x_{v_j} \right).$$

(53)

Here $e$ are horizontal edges connecting blue faces. However we use only alternating horizontal edges while taking the product of the operators of the form $z_{v_i}z_{v_j}$. This results in the identity

$$\prod_{j=1}^f PX_j \prod_{v_i,v_j \in e} z_{v_i}z_{v_j} |s_{Z}\rangle = \prod_{v_i,v_j \in e} z_{v_i}z_{v_j} \prod_{j=1}^f PX_j |s_{Z}\rangle,$$

(54)

which is required in the derivation of Eq. (52). We have used the following identities,

$$PY \perp |s_{Z}\rangle \equiv PX |s_{Z}\rangle,$$

(55)
\[ PY|s_Z\rangle \equiv PX^\perp|s_Z\rangle, \quad (56) \]

on hexagonal faces while deriving Eq. (52). The operators \( PX^\perp \) denote the projectors orthogonal to \( PX \). Thus we find that the ground state or the encoded qubit of the [444]-color code with homogeneous edge configurations are equivalent to those of the original CSS [336]-color code by an LU. The form of the LU is such that the ground state of the former code is an excited state for the latter. Due to this LU the logical Pauli operators of the [444]-color code are obtained by conjugating the logical Pauli’s of the [336]-color code. In particular the logical X’s along two colored lattices can be used to construct the other encoded qubit states spanning the entire code space. In particular this implies that the logical Pauli’s of the canonical [444]-color code coincide with those of the CSS [336]-color codes. Even though the canonical 444-color code is a non-CSS code, its logical Pauli operators are of the homogenous type.

It is also worth noting that though the stabilizers of the two codes are not LU equivalent to each other, their code spaces are still LU equivalent. Furthermore the LU equivalence to the CSS [336]-color code guarantees fault tolerant nature of the code space of the canonical [444]-color code as the logical Clifford operators continue to remain transversal. These statements are not too surprising as the long-range properties of the [444]- and [336]-color codes are expected to coincide. The difference between the two is attributed to the short range spectrum of the two models.

3. Anyon Content

In the CSS [336]-color code electric charges, \( e \) (magnetic fluxes, \( m \)) are those states violating the X (Z) stabilizers, obtained using local string operators acting on the shrunk lattices of a chosen color. The non-CSS nature of the [444]-color codes makes way for a similar narrative with ‘electric charges’ (‘magnetic fluxes’) being those states obtained when the I (II) stabilizer invariance conditions are not satisfied. The ‘dyonic’ excitations are obtained when both the I and the II stabilizer invariance conditions are simultaneously not satisfied, such as for a Y error on a bulk vertex for the CSS [336]-color code. The choice of what we label I and II kind of stabilizer is a convention and the picture remains the same when we make the alternative choice.

Adopting this convention we see that \( e \) and \( m \) excitations are created by acting with string operators made of the Pauli \( x \) and Pauli \( z \) operators on the red shrunk lattice respectively. In a similar manner the \( e \) and \( m \) excitations are created by string operators made of Pauli \( z \) and Pauli \( y \) operators on the blue shrunk lattice. These string operators either commute or anticommute with the stabilizers, obtained using local string operators acting on the shrunk lattices of a chosen color. The non-CSS stabilizers, obtained using local string operators acting on the shrunk lattices of a chosen color. The non-CSS nature of the code makes way for a similar stabilizer invariance to compute the GSD or the dimension of the code space. A set of four independent constraints can be read off from the stabilizers of the code in Fig. 25,

\[
\beta_1 = \prod_{j=1}^{f} r_{1j} \prod_{j=1}^{f} b_{1j}, \quad \beta_2 = \prod_{j=1}^{f} b_{1j} \prod_{j=1}^{f} g_{1j}, \\
\beta_3 = \prod_{j=1}^{f} g_{1j} \prod_{j=1}^{f} r_{1j}, \quad \beta_4 = \prod_{j=1}^{f} r_{1j} \prod_{j=1}^{f} g_{1j} \prod_{j=1}^{f} g_{1j}.
\]

This leads to \(|V| - 2|F| + 4\) encoded qubits which reduces to \(4g\) for a trivalent, tricolorable lattice discretizing a surface of genus \(g\).

2. The Code Space

In this case we can choose the product state \(|s_Z\rangle\) as the seed state that can be projected to the code space.
A ground state is constructed as,

$$|\bar{0}_1\bar{0}_2\bar{0}_3\bar{0}_4\rangle_{[345]} = \prod_{j=1}^{f} P Y_j \prod_{j=1}^{f} P Y_j \prod_{j=1}^{f} P X_j \prod_{j=1}^{f} P X_j |s_z\rangle,$$

$$= \prod_{j=1}^{f} P X_j^{z_j} \prod_{j=1}^{f} P X_j^{z_j} \prod_{j=1}^{f} P X_j^{s_z},$$

$$= U |0_10_20_30_4\rangle_{[336]},$$

where

$$U = \prod_{v_i,v_j \in e} z_{v_i} z_{v_j} \prod_{v_i,v_j \in e} z_{v_i} z_{v_j},$$

is the LU that maps the ground state of the deconfined [345]-color code with homogeneous edge configurations to that of the CSS [336]-color code, and vice versa. Here e and e are alternating horizontal edges connecting red and blue faces respectively (See Fig. 30). This LU equivalence to the CSS [336]-color code ensures that the deconfined [345]-color code supports fault-tolerant computing as well. The remaining ground states or encoded qubits are obtained from the latter. The e and e electric charges are obtained from string operators made of the Pauli z operators. The m and m magnetic fluxes are obtained from string operators made of the Pauli y operators. These string operators act on the ground state $|0_10_20_30_4\rangle_{[345]}$ of Eq. (59) to create the appropriate excitations. As they either commute or anticommute with the LU,

$$U = \prod_{v_i,v_j \in e} z_{v_i} z_{v_j} \prod_{v_i,v_j \in e} z_{v_i} z_{v_j},$$

the resulting excitations are LU equivalent to the excitations of the CSS [336]-color code. In particular, the electric charges of the deconfined [345]-color codes are LU equivalent to the electric charges of the CSS [336]-color code whereas the magnetic fluxes are LU equivalent to the dyons of the CSS [336]-color code.

C. Confined [345]-color codes

There are three types of inequivalent [345] bulk vertices (See Fig. 12) that can be used to build the confined [345]-color codes that are translationally invariant and have homogeneous edge configurations (See Fig. 27). We analyze the model belonging to Class I without loss of any of the features. Unlike the previous models the translational invariance in this system manifests itself in a different manner. Thinking of the hexagonal lattice in terms of columns we see that in the class I model of the confined [345]-color code (See Fig. 27), there is an X and Z column in the stabilizer pair, followed by a Y and Z column which is continued by a column pair that is a mixture of X and Y’s. In the third column we notice that for a face of a given color there are one X and one Y operators similar to what we saw in the canonical [444]-color code. We indicate these column pairs as column 1-1’, column 2-2’ and column 3-3’ to refer to them easily in what follows. The class I confined [345]-color code is shown again in Fig. 31 but this time with the column pairs clearly indicated.

1. Ground State Degeneracy

The dimension of the code space is much harder to compute in this case due to the peculiar nature in which the translational invariance is implemented in this system. Nevertheless by inspecting the stabilizers in Fig. 31 we see that we can pick four independent constraints among the set of stabilizers just by looking at the colors: these are $r_1 b_1$, $r_1 b_{11}$, $b_{11} g_1$ and $b_{11} g_{11}$. Following this color scheme it is quite complicated to write down the stabilizer constraints for the case of the confined [345]-color code and so we request the reader to convince themselves that this is indeed the case by looking at the stabilizer pairs for this system in Fig. 31. We will see indirect
ways of affirming this fact by studying the LU equivalence between the code spaces of the confined [345]- and the CSS [336]-color codes next.

2. The Code Space

The operators occurring in the stabilizer pair in this case is similar to what we saw for the canonical [444]-color code and due to this a homogeneous seed state cannot be projected onto the code space for this system. To rectify this we choose a seed state by taking tensor products of $|0_z\rangle$'s and $|1_z\rangle$'s as we did while analyzing the code space of the canonical [444]-color code. The operators in the column 3 of the stabilizer pairs contain the $X$ and $Y$ operators for each of the colored faces. We choose the seed state such that these faces contain exactly one $|1_z\rangle$ while the remaining are made of $|0_z\rangle$'s. This is achieved by flipping the $|0_z\rangle$'s of the $|s_Z\rangle$ state along the edges in the blue shrunk lattice as shown in Fig. 32. Thus the seed state, $|s\rangle_{[345]}^{\text{confined}}$ is

$$|s\rangle_{[345]}^{\text{confined}} = \prod_{v_i, v_j, v_k \in e} x_{v_i} x_{v_j} x_{v_k} |s_Z\rangle.$$  

This state continues to be a $+1$ eigenstate of the $Z$ stabilizers. The encoded qubit of the confined [345]-color code of class $I$ can now be evaluated as,

$$|0_1 0_2 0_3 0_4\rangle_{[345]}^{\text{confined}} = \left( \prod_{j=1}^{f_1} PX_j \right) \left( \prod_{j=1}^{f_2} PY_j \right) \left( \prod_{j=1}^{f_3} PX_j \right) |s_Z\rangle,$$

where

$$U = \left( \prod_{v_i, v_j, v_k \in e} x_{v_i} x_{v_j} x_{v_k} \right) \left( \prod_{v_i, v_j, v_k \in e} z_{v_i} z_{v_j} z_{v_k} \right).$$

Here the $e$ are horizontal edges in the blue shrunk lattice connecting two column 3’s and the $e$ are horizontal edges in the red shrunk lattice connecting two column 2’s. The $f_i$, $f_i$ and $f_i$ for $i \in \{1, 2, 3\}$ are the total number of red, blue and green faces in columns 1,2,3 respectively. The state in Eq. (63) can also be viewed as an excitation of the [336]-color code. However unlike the previous cases this excitation is an instance where sets of three stabilizers of the [336]-color code are excited and hence are not deconfined anyons where either two or four stabilizers of a given color are excited. This is not surprising as we

![Figure 31. The translationally invariant confined [345]-color code with the column pairs, $j - j'$ with $j \in \{1, 2, 3\}$, indicated.](image)

![Figure 32. The support of the edges along the blue shrunk lattice, connecting column 3's. The sites marked with blue crosses are filled with the $|1_z\rangle$ state in the tensor product with the remaining sites filled with $|0_z\rangle$.](image)
are dealing with a confined [345]-color code here. The remaining encoded qubits are obtained by acting the winding operators on the \(|0_10_20_4\rangle_{[345],\text{confined}}\) state. Notice that the winding operators either commute or anticommute with the LU in Eq. 64 and hence all the encoded qubits remain LU equivalent to the code space of the CSS [336]-color code up to a negative sign.

3. Anyon Content

In this case the elementary strings supported on two sites of a chosen shrunk lattice, either \(x_v x_v, y_v y_v\) or \(z_v z_v\), excite either three or two of the stabilizers in the stabilizer pair. As a result of this extending strings along certain directions can increase or decrease the energy. This is fundamentally different from the deconfined anyons encountered in the earlier cases including the deconfined [345]-color code. However as the string operators creating these excitations by acting on the \(|0_10_20_4\rangle_{[345],\text{confined}}\) either commute or anticommute with the LU, \(U\) in Eq. (64), the resulting excited states are still equivalent to the anyons of the CSS [336]-color code.

VII. TRIANGULAR NON-CSS COLOR CODES

We will now discuss the non-CSS color codes on finite lattices that include boundary and corner vertices apart from the bulk vertices. The 7-qubit Steane code is the simplest example of a color code on a finite lattice, where the seven physical qubits are conveniently placed in the geometry of a triangle. The framework introduced here allows the construction of 7-qubit non-CSS codes that are not connected to the Steane code by an LU.

There is a single bulk vertex at the center of the 7-qubit triangle implying that there are exactly two more inequivalent Steane codes, one for the [444] bulk vertex and the other for the [345] bulk vertex. The three inequivalent [345] bulk vertex configurations in Fig. 12 are mapped to each other by a global rotation. We also obtain non-CSS [336]-Steane codes that are equivalent to the canonical CSS Steane code by an LU. We include these for completion.

Due to the non-CSS nature of the code we denote the pair of stabilizers for every face as I and II along with the color of the face. It is easily seen that each of these operators square to one, generating a \(\mathbb{Z}_2\) and together they form a subgroup of the 7-qubit Pauli group. A little bit of work also shows that -1 is not generated from these stabilizers. Using the six stabilizers we construct projectors, for example \(P_{I} = \frac{1 + r_I}{2}\), where 1 is the identity operator acting on the 7-qubits. The code space is spanned by the ground states of the Hamiltonian formed out of these projectors as in the case of the original color code,

\[
H = -(P_{I} + P_{II} + P_{bI} + P_{bII} + P_{gI} + P_{gII}).
\]

(65)

As the six stabilizers of this code are independent the Hamiltonian in Eq. (65) has two ground states or it encodes a single logical qubit, \(\{|0_L\}, |1_L\}\).

A. [444]-Steane code

a. Homogeneous edge configuration - The stabilizers for the canonical [444]-7-qubit Steane code are shown in Fig. 33 (a). By inspection we find that the logical Pauli’s for this code remain the same as the original [336]-Steane code,

\[
X_L = x_1 x_2 x_3 x_4 x_5 x_6 x_7,
\]
\[ Z_L = z_1 z_2 z_3 z_4 z_5 z_6 z_7, \]
\[ Y_L = -y_1 y_2 y_3 y_4 y_5 y_6 y_7. \]  
(66)

A more systematic way of arriving at these expressions is as follows. Consider the symmetries of the stabilizers of the [444]-Steane code, that are those operators that cannot be written as a product of the stabilizers. We find distance-three operators that have support along the boundaries of the triangle and another set of distance-three operators that have support along the diagonals of the triangle. Explicitly,
\[ \{x_1 x_2 x_3, x_3 x_6 x_5, x_5 x_4 x_1\}, \]  
(67)

and
\[ \{x_1 x_7 x_6, x_3 x_7 x_4, x_5 x_7 x_2\}, \]  
(68)

are two sets of distance-three symmetries along the boundaries and diagonals respectively. It is easy to see that any operator in these two sets are equivalent to the logical Pauli, \(X_L\) (See Eq. (66)) up to a multiplication by a stabilizer of the [444]-Steane code with unmixed edge configurations. The logical Pauli’s in Eq. (66) are chosen in such a way that they have support on all the seven qubits. They are nevertheless stabilizer equivalent to appropriate weight-three symmetries. A similar method can be used to find the logical Pauli, \(Z_L\) and hence \(Y_L\).

Having obtained the logical Pauli’s the ground state can now be constructed by projecting the eigenstate of the \(Z_L\) operator using the stabilizers, \(r_I, r_{II}, b_I, g_{II}\). A simple calculation shows that this state coincides with that of the original Steane code,
\[ |0_L\rangle = \frac{1}{8} \left[ |0_1 0_2 0_3 0_4 0_5 0_6 0_7\rangle + |0_1 0_2 0_3 1_4 1_5 1_6 1_7\rangle + |1_1 1_2 1_3 0_4 0_5 1_6 0_7\rangle + |1_1 1_2 1_3 1_4 0_5 1_6 0_7\rangle + |1_2 1_3 1_4 0_5 0_6 1_7\rangle + |1_2 1_3 1_4 1_5 0_6 1_7\rangle \right]. \]  
(69)

An easy way to understand this is by observing that the action of the \(Y\) stabilizer on the eigenstate of \(Z_L\) is the same as the \(X\) stabilizer as each of them has the weight four. However this is not true when the stabilizers have the weight six as then we have \(i^6 = -1\) and so the \(Y\) stabilizer on a hexagon acts as \(-X\) stabilizer on the eigenstate of \(Z_L\). This will lead to important differences when we consider triangle codes of larger sizes. However for the simplest case of the 7-qubit triangle the \(Y\) and \(X\) stabilizers have the same action on the eigenstates of \(Z_L\).

The other ground state for the [444]-Steane code is obtained as \(|1_L\rangle = \chi_L|0_L\rangle\). Thus the codewords of the [444]-Steane code coincide with that of the original Steane code or the [336]-Steane code with unmixed edge configurations even though the two sets of stabilizers are not equivalent to each other by an LU. As a consequence the logical Hadamard, phase gate and CNOT are unchanged as well. Thus the [444]-Steane code provides a set of stabilizers that supports fault-tolerant computation.

\[ b. \text{Mixed edge configurations} \] - The situation is modified for the [444]-Steane codes with partially and fully mixed edge configurations (See Figs. 33 (b), (c)). The code space, the logical Pauli’s and Cliffords see a change now but can be mapped to the canonical [444]-Steane code by the LU’s, \(H_2 S_4 (SH)^3_6\) and \((SH)_2 (HS)^3_4 (SH)_6\) respectively.

It is still illustrative to construct the logical operators for the mixed [444]-Steane codes from the ground up regardless of their equivalence to the canonical [444]-Steane code. This helps in seeing the equivalence class of the logical Pauli’s and inspecting their mixed nature.

1. The logical operators and the code space

We proceed by identifying the symmetries of this Hamiltonian that are not generated by the stabilizers. As there is just a single encoded logical qubit we expect to find just a single set of the logical Pauli’s \(\{X_L, Z_L, Y_L\}\), supported on all the seven qubits.

\[ a. \text{Symmetries} \] - From the geometry of the seven qubit system the symmetries with support exhausting all the seven qubits can be obtained by considering the symmetries along the boundaries (edges of the triangle) and the diagonals of the triangle (symmetries that include support on the qubit 7). For the [444]-Steane code with partially mixed edge configurations (See Fig. 33 (b)), the boundary symmetries are given by
\[ \{x_1 x_2 x_3, x_5 x_6 x_3, x_1 x_4 x_5\} \]  
(70)

and the diagonal symmetries are given by
\[ \{y_1 y_2 y_3, y_5 y_6 y_3, y_1 x_4 y_5\} \]  
(71)

\[ \{z_1 x_2 z_3, z_5 y_6 z_3, z_1 z_4 z_5\}, \]  
(72)

\[ \{z_1 x_7 y_6, z_5 x_7 z_2, z_3 z_7 z_4\}. \]  
(73)

Notice that each of the symmetries are of weight three and all the operators inside a given set commute with each other. On the other hand operators belonging to different sets anticommute with each other. We also note that the operators in Eqs. (70) and (73) commute with each other and do the operators in Eqs. (71) and (74) and the operators in Eqs. (72) and (75). Furthermore, we can check that the operators within a set are related to each other, up to a fourth root of unity, by the stabilizers of the model. For example,
\[ x_1 z_2 x_3 \sim x_1 z_2 x_3 (r_I) \sim x_3 x_7 y_4, \]  
(76)

\[ x_1 z_2 x_3 \sim x_1 z_2 x_3 (r_I b_I) \sim x_3 x_6 x_3. \]  
(77)

The logical Paulis, \(X_L\) and \(Z_L\), can be written as
\[ X'_L = x_1 x_2 x_3 y_4 x_5 x_6 x_7 \]  
(78)

\[ Z'_L = z_1 x_2 z_3 . z_4 z_5 y_6 z_7. \]  
(79)

This choice is arbitrary and other choices just lead to rotated code states.
b. Code Space - To construct the encoded qubit consider the +1 eigenstate of the logical Pauli operators $Z_L^+$ which are $|s_{z'}\rangle = |z_1, z_2, z_3, z_4, z_5, z_6, z_7\rangle$. We have used the +1 eigenstates of the Pauli operators, $x, y$ and $z$ as $|\phi_x\rangle = \frac{1}{\sqrt{2}}|0 \rangle + |1 \rangle$, $|\phi_y\rangle = \frac{1}{\sqrt{2}}|0 \rangle - |1 \rangle$ and $|\phi_z\rangle \equiv |0 \rangle$ respectively. The encoded qubit is then

$$|0_L\rangle = P_{y_1} P_{y_11} P_{b_1} P_{g_11} |s_{z'}\rangle, \quad (80)$$

$$|1_L\rangle = |x_L\rangle |0_L\rangle. \quad (81)$$

Alternatively we can use the LU to the canonical [444]-Steane code to construct the logical qubit for the partially mixed model,

$$|0_L\rangle = H_2 S_1^3 (S H S^3)^6 |0_L\rangle, \quad |1_L\rangle = H_2 S_1^3 (S H S^3)^6 |1_L\rangle, \quad (82)$$

where $|0_L\rangle$ in Eq. (69), and $|1_L\rangle$ are the encoded qubits of the original $[336]$-Steane code.

c. Logical Cliffords - Using the LU, $H_2 S_1 (S H S^3)_6$ between the canonical [444]-Steane code and the partially mixed [444]-Steane code we obtain the logical Cliffords as

$$H_L = H_1 H_2 H_3 (S H S^3)_4 H_5 (S H S^3 H S H S^3)_6 H_7, \quad (83)$$

and

$$S_L = S_1 (S H S H)_2 S_3 S_4 S_5 (S H S S^3)_6 S_7, \quad (84)$$

where $t = S H S^3$ and $CNOT_L = CNOT_{11} \cdots CNOT_{77}$. The primed indices denote another copy of the Steane code. Note that the transversal nature of the Pauli operators is not affected across the various models.

2. The fully mixed [444]-Steane code

The stabilizers of the [444]-Steane code with fully mixed edge configurations (See Fig. 33 (c)) are equivalent to the canonical [444]-Steane code by the LU, $(S H)_2 (H S^3)_4 (H S)_6$. The boundary symmetries in this case are

$$\{x_1 z_2 x_3, x_5 z_6 x_3, x_1 y_4 x_5\} \quad (86)$$

$$\{y_1 x_2 y_3, y_5 x_6 y_3, y_1 z_4 y_3\} \quad (87)$$

$$\{z_1 y_2 z_3, z_5 y_6 z_3, z_1 x_4 z_5\}, \quad (88)$$

and the diagonal symmetries are given by

$$\{x_1 z_7 z_6, x_5 z_7 z_2, x_3 z_7 y_4\} \quad (89)$$

$$\{y_1 y_7 x_6, y_5 y_7 x_2, y_3 y_7 z_4\} \quad (90)$$

$$\{z_1 z_7 y_6, z_5 z_7 y_2, z_3 z_7 x_4\}. \quad (91)$$

As in the previous case the operators within a set commute and those belonging to different sets anticommute. The operators within a single set are related to each other by the action of the stabilizers of the code in Fig. 33 (c).

Using the symmetries we build the logical Pauli operators with support on all seven qubits as,

$$X'_{L'} = x_1 z_2 x_3 y_4 x_5 z_6 x_7 \quad (92)$$

$$Z''_{L'} = z_1 y_2 z_3 x_4 z_5 y_6 z_7. \quad (93)$$

The ground states can be constructed by projecting the +1 eigenstate of $Z''_L$ using the stabilizers of the code in Fig. 33 (c). Using the LU the logical qubit in this case is written as

$$|0_L\rangle = (H S^3)_2 (S H)_4 (H S^3)_6 |0_L\rangle, \quad (94)$$

$$|1_L\rangle = (H S^3)_2 (S H)_4 (H S^3)_6 |1_L\rangle, \quad (95)$$

where $|0_L\rangle$ is the original Steane code, Eq. (69) and this coincides with the alternate construction of acting the projects built out of the stabilizers of the [444]-Steane code with fully mixed edge configurations on the +1 eigenstate of $Z''_L$. The logical Cliffords are

$$H''_L = H_1 (S H S^3)_2 H_3 (H S^3 H S H)_4 H_5 (S H S^3)_6 H_7, \quad (96)$$

$$S''_L = S_1 (S H S S^3)_2 S_3 (H S H)_4 S_5 (S H S S^3)_6 S_7, \quad (97)$$

and the new $CNOT''_L$ gate is obtained by conjugating the old one with $(S H)_2 (H S^3)_4 (S H)_6 (S H)_2 (S H^3)_4 (S H)_6$.

B. [345]-Steane code

The stabilizers for the canonical, partially and fully mixed, [345]-7-qubit Steane code are shown in Fig. 34. Though there are three inequivalent [345] bulk vertices (See Fig. 12), they can be mapped into each other by a global rotation of all the stabilizers of the code. This is a consequence of the fact there is just a single bulk vertex to contend with in the Steane code. Hence we pick one of them for further consideration without a loss of generality. The Hamiltonian is built out of the projects corresponding to these stabilizers as in the [444]-Steane code. The dimension of the code space is once again 2 as all the stabilizers are independent. To construct the logical qubit we once again identify the symmetries and the logical Paulis. We note that for canonical [345]-Steane code in Fig. 34 (a), the logical Pauli operators are the same as the original [336]-Steane code. The logical qubit is also unchanged and is given by $|0_L\rangle$ and $|1_L\rangle$ in Eq. (69). We find a change in the expressions for the logical operators and the logical qubits when we use partially and fully mixed edge configurations as shown in Figs. 34 (b), (c). However these codes are equivalent to the canonical [345]-Steane code by LU’s $(S H)_2 (S H)_3 S_1 (H S^3)_5 (S H S^3)_6$ and $(S H)_2 (H S^3)_4 (S H)_5 (H S^3)_6$ respectively.
As in the case of the [444]-Steane code, the symmetries and combined support covers all the seven physical qubits, edge configurations are either (a) homogeneous, (b) partially mixed 

Indices of the physical qubits are shown with the 7th qubit.

\[
\begin{align*}
X_L' &= z_1z_2z_3y_4y_5x_6x_7 \\
Z_L' &= y_1x_2y_3z_4z_5y_6z_7.
\end{align*}
\]

The logical qubit is constructed using the LU as

\[
|0_L'\rangle = (HS^3)_1H_2(HS^3)_3S^3_4(SH)_5(SHS^3)_6|0_L\rangle \tag{106}
\]

\[
|1_L'\rangle = (HS^3)_1H_2(HS^3)_3S^3_4(SH)_5(SHS^3)_6|1_L\rangle. \tag{107}
\]

The logical Pauli’s are then,

\[
\begin{align*}
X_L'' &= x_1z_2x_3y_4z_5y_6x_7 \\
Z_L'' &= y_1z_2y_3z_4y_5z_6x_7.
\end{align*}
\]

The logical qubits are obtained by conjugating the transversal Cliffords of the canonical [345]-Steane code by the LU, \((SH)_1H_2(SH)_3S_4(HS^3)_5(SHS^3)_6\).

2. The fully mixed [345]-Steane code

The boundary and the diagonal symmetries, whose combined support covers all the seven physical qubits, are given by,

\[
\begin{align*}
\{x_1y_2x_3, z_5y_6x_3, x_1y_4z_5\} \quad (108) \\
\{y_1z_2y_3, y_5x_6z_3, z_1x_4y_5\} \quad (109) \\
\{y_1x_2y_3, x_5z_6y_3, y_1z_4x_5\}, \quad (110)
\end{align*}
\]

and

\[
\begin{align*}
\{x_1x_7y_6, z_5x_7z_2, x_3x_7y_4\} \quad (111) \\
\{z_1x_7z_6, y_5x_7y_2, z_3z_7x_4\} \quad (112) \\
\{y_1y_7z_6, x_5y_7x_2, y_3y_7z_4\}. \quad (113)
\end{align*}
\]

The logical Pauli’s are then,

\[
\begin{align*}
X_L'' &= x_1z_2x_3y_4z_5y_6x_7 \\
Z_L'' &= y_1z_2y_3z_4y_5z_6x_7.
\end{align*}
\]

The logical qubit is constructed using the LU as

\[
\begin{align*}
|0_L''\rangle &= (HS^3)_2(SH)_4(HS^3)_5(SH)_6 |0_L\rangle, \tag{116} \\
|1_L''\rangle &= (HS^3)_2(SH)_4(HS^3)_5(SH)_6 |1_L\rangle. \tag{117}
\end{align*}
\]

The logical Cliffords are obtained by conjugating the transversal Cliffords of the canonical [345]-Steane code by the LU, \((SH)_2(HS^3)_4(SH)_5(HS^3)_6\).

C. [336]-Steane code

For completion we include the canonical, partially and fully mixed [336]-Steane codes in Fig. 35. The canonical [336]-Steane code in Fig. 35 (a) is the original Steane code and its logical qubit and operators are well known. We will discuss the [336]-Steane codes with partially and fully mixed edge configurations that are equivalent to the canonical [336]-Steane code by the LU’s, \((HS^3)^3_4(SHS^3)^3_2((SH)_5)_6\) and \((SH)_2(HS^3)_4(SH)_5(HS^3)_6\) respectively.
As in the earlier cases, the symmetries within each tries in the other. The stabilizers of the left triangle pick up the suffix $I$ indexing the lone bulk vertex. The stabilizers of the left triangle and those in the right, $II$. The edge configurations are either (a) homogeneous, (b) partially mixed or (c) fully mixed.

1. The partially mixed [336]-Steane code

The boundary and the diagonal symmetries, whose combined support covers all the seven physical qubits, are given by,

\[
\{y_1x_2y_3, \ z_5x_6y_3, \ y_1y_4z_5\} \quad (118) \\
\{x_1y_2x_3, \ y_5x_6x_3, \ x_1z_4y_5\} \quad (119) \\
\{z_1z_2z_3, \ x_5y_6z_3, \ z_1x_4x_5\}, \quad (120)
\]

and

\[
\{y_1x_7z_6, \ z_5x_7x_2, \ y_3x_7y_4\} \quad (121) \\
\{x_1x_7y_6, \ y_5x_7y_2, \ x_3z_7z_4\} \quad (122) \\
\{z_1y_7y_6, \ x_5y_7z_2, \ z_3y_7x_4\}. \quad (123)
\]

As in the earlier cases, the symmetries within each $\{\}$ mutually commute and anticommute with the symmetries in the other $\{\}$. Also the symmetries in Eqs. (118) and (121), those in Eqs. (119) and (122) and those in Eqs. (120) and (123) mutually commute. This results in the logical Pauli’s that read as

\[
X_L' = y_1x_2y_3y_4z_5z_6x_7 \\
Z_L' = x_1y_2x_3z_4y_5x_6z_7. \quad (124)
\]

The logical qubit is constructed using the LU as

\[
|0_L'\rangle = (SH)_1(SHS^3)_2(SH)_3S^3_1(SHS^3)_5H_6 \ |0_L\rangle, \quad (126) \\
|1_L'\rangle = (SH)_1(SHS^3)_2(SHS^3)_5S^3_4(SHS^3)_5H_6 \ |1_L\rangle. \quad (127)
\]

The logical Cliffords are obtained by conjugating the transversal Clifford gates of the canonical [336]-Steane code by the LU, $(HS^3)_1(SHS^3)_2(HS^3)_3S^3_4(SHS^3)_5H_6$.

2. The fully mixed [336]-Steane code

The boundary and the diagonal symmetries, whose combined support covers all the seven physical qubits, are given by,

\[
\{x_1z_2x_3, \ z_5y_6x_3, \ x_1y_4z_5\} \quad (128) \\
\{z_1y_2z_3, \ y_5x_6z_3, \ z_1x_4y_5\} \quad (129) \\
\{y_1x_2y_3, \ x_5z_6y_3, \ y_1z_4x_5\}, \quad (130)
\]

and

\[
\{x_1x_7y_6, \ z_5x_7z_2, \ x_3x_7y_4\} \quad (131) \\
\{z_1z_7y_6, \ y_5z_7y_2, \ z_3z_7z_4\} \quad (132) \\
\{y_1y_7z_6, \ x_5y_7x_2, \ y_3y_7z_4\}. \quad (133)
\]

As in the earlier cases, the symmetries within each $\{\}$ mutually commute and anticommute with the symmetries in the other $\{\}. Also the symmetries in Eqs. (128) and (131), those in Eqs. (129) and (132), and those in Eqs. (130) and (133) mutually commute. This results in the logical Pauli’s that read as

\[
X_L'' = x_1z_2x_3y_4z_5y_6x_7 \\
Z_L'' = z_1y_2z_3x_4y_5x_6z_7. \quad (134)
\]

The logical qubit is constructed using the LU as

\[
|0_L''\rangle = (HS^3)_2(SH)_4(HS^3)_5(SH)_6 \ |0_L\rangle, \quad (136) \\
|1_L''\rangle = (HS^3)_2(SH)_4(HS^3)_5(SH)_6 \ |1_L\rangle. \quad (137)
\]

The logical Cliffords are obtained by conjugating the transversal Clifford gates of the canonical [336]-Steane code by the LU, $(SH)_2(HS^3)_4(SH)_5(HS^3)_6$.

D. The 19-qubit code

The triangle geometry of the 7-qubit Steane code can be cut out from the hexagon lattice such that we do not include any full hexagon. A natural generalization of this is cutting out a larger triangle such that we include
Along with the boundary symmetries, each of which are exactly one full hexagon of each color resulting in the 19-qubit code (See Fig. 36). The 7-qubit and 19-qubit codes are examples of distance-three and distance-five codes in the triangle code family.

There are three faces for each color and thus six stabilizers for each color summing up to 18 independent stabilizers that results in a single encoded qubit. The stabilizers are denoted \((r_i)_j, (b_i)_j, (g_i)_j\) for \(i \in \{a, b, c\}\) and \(j \in \{I, II\}\). The stabilizers generate an abelian subgroup of the 19-qubit Pauli group and the projectors built out of them is used to construct the commuting projector Hamiltonian just as in the previous cases. Before we analyze the [345]- and [444]-19 qubit codes we first review the case of the original CSS [336]-19 qubit codes. We construct the code words or the logical qubit and the appropriate logical Pauli and Clifford operations.

1. [336]-19 qubit code

The CSS version of the 19 qubit codes are stabilized by the \(X\) and \(Z\) operators for each face of the 19 qubit triangle. The logical Pauli’s are given by

\[
X_L = x_1 \cdots x_{19}, \quad Z_L = z_1 \cdots z_{19}.
\]

(138)

As there is just one encoded qubit we expect to find just a single set of logical Pauli operators. These can be systematically obtained by identifying the symmetries of the stabilizers just as in the case of the 7 qubit Steane codes. In this case we consider the symmetries along the boundaries of the triangle and this covers the sites 1 through 12 (See Fig. 36). Next we consider symmetries that are stabilizer equivalent to the boundary symmetries by beginning at a corner vertex and then we cut through the triangle spanning bulk vertices and end on a boundary vertex opposing the chosen corner vertex. For example in the green lattice of Fig. 36, three such symmetries exhausting all the bulk vertices are given by, \(x_1 x_{13} x_{14} x_9 x_6, x_1 x_{13} x_{14} x_{18} x_7\) and \(x_1 x_{13} x_{14} x_{15} x_{16} x_{17} x_8\). Note that these symmetries are at least of distance five. Along with the boundary symmetries, each of which are distance-five symmetries, we cover all the sites of the 19 qubit triangle.

The encoded qubit is constructed by projecting, \(|s_Z\rangle\) the eigenstate of the \(Z_L\) logical with the \(X\) stabilizers,

\[
|0_L\rangle = \prod_{j \in \{a, b, c\}} PX_j PX_j PX_j |s_Z\rangle, \quad |1_L\rangle = X_L |0_L\rangle.
\]

(139)

a. **Logical Hadamard, \(H_L\)** - The transversal logical Hadamard in this case is given by

\[
H_L = H_1 \cdots H_{19}.
\]

(140)

This is verified to be the right expression by checking its action on the logical qubit as,

\[
H_L |0_L\rangle = H_L \prod_{j \in \{a, b, c\}} PX_j PX_j PX_j |s_Z\rangle, \quad = \prod_{j \in \{a, b, c\}} PZ_j PZ_j PZ_j |s_X\rangle,
\]

(141)

where \(|s_X\rangle\) is the \(+1\) eigenstate of \(X_L\). This state is precisely the dual state and is given by \(\frac{|0_L\rangle + |1_L\rangle}{\sqrt{2}}\). It is easily seen that \(H_L |1_L\rangle = \frac{|0_L\rangle - |1_L\rangle}{\sqrt{2}}\). Moreover the logical Hadamard consistently acts on the logical Pauli’s by interchanging \(X_L\) and \(Z_L\) under conjugation.

b. **Logical Phase Gate, \(S_L\)** - Obtaining the transversal logical phase gate is a bit more subtle as a naive choice of \(S_1 \cdots S_{19}\) does not have the right action on the logical qubit. This is seen as follows,

\[
S_L |0_L\rangle = S_L \prod_{j \in \{a, b, c\}} PX_j PX_j PX_j |s_Z\rangle, \quad \quad = \prod_{j \in \{a, b, c\}} PY_j PY_j PY_j |s_Z\rangle,
\]

(142)

using \(SXS^3 = Y\). However the resulting state is not the same as \(|0_L\rangle\) as \(PY\) supported on a hexagon acts on \(|s_Z\rangle\) as \(PX^3\) or the orthogonal complement of \(PX\). This shows that the naive choice for the transversal logical phase gate is not the right one. To fix this we use \(S^3\) on half of the sites for each hexagon such that the rest of the stabilizers supported on the quadrilaterals transform as before. This will ensure that the \(PX\) operators on each hexagon transforms to \(PY^3\) which acts on \(|s_Z\rangle\) in the same way as \(PX\). Thus the consistent choice for the logical phase gate is given by

\[
S_L = \prod_{j \in B} S^3 \prod_{j \notin B} S,
\]

(143)

where the set comprised of the sites \(B = \{3, 4, 7, 8, 11, 12, 13, 17, 19\}\). This operator has the right action on \(X_L\) transforming it to \(Y_L\) under conjugation and commutes with \(Z_L\) leaving it invariant.
c. **Logical CNOT Gate, CNOT_L** - The CNOT gate is a two qubit gate which performs the NOT operation with the first qubit as a control qubit. Explicitly it acts as a Pauli X on the second qubit if the first qubit is in the |1⟩ state. As an operator this translates to

\[
\text{CNOT}_{12} = \frac{1}{2} [1 + Z_1 + X_2 - Z_1 X_2],
\]

(144)

where 1 and 2 index the two qubits. Under conjugation by CNOT the Pauli operators on the two qubits transform as

\[
\begin{align*}
X_1 &\rightarrow X_1 X_2, \quad X_2 \rightarrow X_2, \\
Z_1 &\rightarrow Z_1, \quad Z_2 \rightarrow Z_1 Z_2.
\end{align*}
\]

(145)

These expressions will be useful in constructing the logical CNOT_L on the encoded qubits of the [336]-19 qubit code. We pick the naive choice that is,

\[
\text{CNOT}_L = \prod_{j=1}^{19} (\text{CNOT})_{jj'},
\]

(146)

the unprimed and the primed indices denote the two copies of the [336]-19 qubit codes. To verify that this choice indeed does the job, we verify its action on two copies of the encoded qubits. Explicitly,

\[
\text{CNOT}_L |0_L, 0_L⟩ = \prod_{j \in \{a,b,c\}} PX_j PX_j PX_j
\]

\[
\prod_{j \in \{a,b,c\}} PX_j PX_j PX_j' |s_Z⟩ \otimes |s_Z⟩
\]

\[
= \prod_{j, j' \in \{a,b,c\}} PX_j X_j' PX_j X_j' PX_j X_j
\]

\[
\prod_{j \in \{a,b,c\}} PX_j PX_j PX_j' |s_Z⟩ \otimes |s_Z⟩
\]

(147)

which coincides with |0_L, 0_L⟩ as PX_j X_j' PX_j = PX_j PX_j'. Clearly it leaves |0_L, 1_L⟩ also unchanged as |0_L, 1_L⟩ = X'_L |0_L, 0_L⟩ and CNOT_L does not transform the Pauli X operator acting on the second copy. On the other hand when the control qubit is |1_L⟩, the CNOT_L operator changes the second encoded qubit. For example,

\[
\text{CNOT}_L |1_L, 0_L⟩ = \text{CNOT}_L X_L |0_L, 0_L⟩,
\]

\[
= X_L X'_L \text{CNOT}_L |0_L, 0_L⟩,
\]

\[
= |1_L, 1_L⟩.
\]

(148)

It is easy to see that CNOT_L |1_L, 1_L⟩ = |1_L, 0_L⟩ as CNOT^2_L = 1. Thus the naive choice for the transversal CNOT_L gate acts in the desired manner on both the logical qubit and the logical Pauli operators.

We have thus established the well known fact that the CSS [336]-19 qubit code supports fault-tolerant computation. Next we proceed to the construction of the logical Pauli’s, the logical qubit and the logical Clifford’s for both the [345]- and [444]-19 qubit codes. We will see that the code words in each of these cases can be mapped to the code words of the CSS [336]-19 qubit code by an LU.

2. **[345]-19 qubit code**

We begin with the [345]-19 qubit codes. Unlike the 7-qubit triangles there are more bulk vertices in the 19-qubit triangles which implies there are more possibilities for inequivalent [345]-triangle codes. More precisely there are three inequivalent [345]-triangle codes corresponding to the inequivalent bulk vertex configurations in Fig. 12.

And for each of these configurations we can build either a deconfined or a confined [345]-19 qubit code.

a. **Deconfined [345]-19 qubit code** - Among the three possible inequivalent [345]-triangle vertex types (see Fig. 12), we discuss just one of them with homogeneous edge configurations. We pick the case where every red face has one Y and one Z stabilizers whereas each of the blue and green faces have one X and one Z stabilizers. To construct the code words we begin with the logical Pauli operators. These operators coincide with those of the CSS [336]-19 qubit codes, that is X_L = x_1 \cdots x_{19} and Z_L = z_1 \cdots z_{19}. Moving on to the construction of the logical qubit we project the |s_Z⟩, the +1 eigenstate of the Z_L operator,

\[
|0_L⟩ = \prod_{j \in \{a,b,c\}} PY_j PX_j PX_j' |s_Z⟩.
\]

(149)

The projectors on the red faces differ from the projectors used in the construction of the logical qubit in the [336]-19 qubit code. As seen earlier the PY_a projector on the red hexagon acts as PX^+_a on the state |s_Z⟩, whereas the PY_b and PY_c on the quadrilaterals act as PX_b and PX_c on the state |s_Z⟩. With the Y projectors reducing in this manner the code word becomes,

\[
|0_L⟩ = PX^+_a PX_b PX_c \prod_{j \in \{a,b,c\}} PX_j PX_j |s_Z⟩,
\]

\[
= z_1 z_{12} \prod_{j \in \{a,b,c\}} PX_j PX_j PX_j |s_Z⟩,
\]

\[
= z_1 z_{12} |0_L⟩,
\]

(150)

with |0_L⟩ being the logical qubit of the CSS [336]-19 qubit code. Note that z_1 z_{12} is not the unique choice that flips PX_a to PX^+_a. Any operator that is equivalent to z_1 z_{12} up to a multiplication by a logical Pauli Z also has the same effect. The state |0_L⟩ can also be viewed as an excited state in the spectrum of the CSS [336]-19 qubit Hamiltonian.

The LU equivalence of the [336]- and the deconfined [345]-19 qubit codes implies that the latter supports fault-tolerant computation as well. The [345]-19 qubit codes with partially and fully mixed edge configurations
are LU equivalent to the one just discussed with homogene- 

geneous edge configurations. Hence they are naturally 

fault-tolerant in nature.

6. Confined [345]-19 qubit code - The stabilizer 

pair for a confined [345]-19 qubit code is shown in Fig. 

37. The logical Pauli operators continue to be the same 

as in the previous cases namely, 

\[ X_L = x_1 \cdots x_{19} \quad \text{and} \quad Z_L = z_1 \cdots z_{19}. \]

The encoded qubit is constructed as 

\[ \langle \tilde{0}_L \rangle = PY_a PY_a PX_a PY_b PX_b PX_c \]

\[ PY_b PY_c PX_Y PX_b PY_Y PX_C |s_Z\rangle. \]  

As we saw earlier the \( Y \) projectors on the quadrilaterals 

act in the same way as the \( X \) projectors on the \( |s_Z\rangle \) state. 

The key difference arises in the action of the \( Y \) projectors 

on the red and green hexagons on the \( |s_Z\rangle \) state. We have 

seen that they act as \( PX^\perp \) on \( |s_Z\rangle \). As a result the \( \tilde{0}_L \) reduces to 

\[ \langle \tilde{0}_L \rangle = (z_1 z_{12}) (z_4 z_5) |0_L\rangle, \]  

with \( |0_L\rangle \) being the code word for the [336]-19 qubit code, 

and thereby establishing the LU equivalence between the 

logical qubit of the confined [345]-19 qubit code and the 

[336]-19 qubit code.

3. [444]-19 qubit code

Consider the canonical [444]-19 qubit code where the 

edge configurations are homogeneous. The red stabilizers 

are made of the \( X \) and \( Y \) operators, the blue faces with 

the \( Y \) and \( Z \) operators and finally the green faces with 

the \( Z \) and \( X \) operators.

It is easy to see that all the symmetries, whose com- 
bined support spans all the 19 qubit locations, are of 
the homogeneous type, that is they are composed of the 
same Pauli operator. This results in logical Pauli’s 
of the homogeneous type, namely the \( X_L \) and the \( Z_L \) 
operators. However if we construct the logical qubit, 
by projecting the +1 eigenstate of either of the logical 
Pauli’s using the projectors in the Hamiltonian, we 
obtain the 0 state as these eigenstates are orthogonal to 
the code space. This can be checked explicitly by evaluating 
\( (PX_a)^1 (PY_a)^1 |s_Z\rangle \), where \( |s_Z\rangle \) is the +1 eigenstate of 
\( Z_L \). Hence we need to make a different choice for the 

seed state that generates the ground state or the logical 

cubit. Instead the state 

\[ |s\rangle = |z_1 z_2 \cdots z_{11} z_{12} z_{13} \cdots z_{19}\rangle, \]

\[ = x_1 x_{12} |s_Z\rangle \]  

has a non-zero projection to the code space and can be 

used to construct the \( \text{logical} \) qubit, \( |0''_L\rangle \), with the 

projectors, 

\[ \prod_{i \in \{a,b,c\}} (PX_i)(PY_i)(PY_i)(PX_i). \]

The logical qubit, \( |0''_L\rangle \) then reduces to, 

\[ |0''_L\rangle = \prod_{i \in \{a,b,c\}} (PX_i)(PY_i)(PY_i)(PX_i) x_1 x_{12} |s_Z\rangle, \]

\[ = x_1 x_{12} (PX_a)(PY_a)(PY_a)(PX_a) \]

\[ \prod_{i \in \{b,c\}} (PX_i)(PY_i)(PY_i)(PX_i) |s_Z\rangle \]

\[ = (PX_a)(PX_a)(PX_a) \]

\[ \prod_{i \in \{b,c\}} (PX_i)(PX_i)(PX_i) |s_Z\rangle \]

\[ = (x_1 x_{12})(z_2 z_9) |0_L\rangle, \]  

establishing the LU equivalence between the code word of 

the [444]-19 qubit code and the [336]-19 qubit code, \( |0_L\rangle \). 

The logical, \( |1''_L\rangle \) is obtained as, \( |1''_L\rangle = X_L |0''_L\rangle \). The 

LU equivalence ensures that the [444]-19 qubit supports 
fault-tolerant computation.

The [444]-19 qubit codes with partially and fully mixed 

dge configurations are equivalent to the canonical [444]- 

19 qubit code by LU’s. The construction of these codes 

follow the algorithm developed earlier or alternatively 

they can be cut out from the corresponding hexagon 

codes in Fig. 21.

VIII. EXAMPLE : A [444]-COLOR CODE WITH 

FULLY MIXED EDGE CONFIGURATIONS - 

MIXED LOGICALS

One of the motivations for studying non-CSS color 
codes is the potential presence of logical Pauli’s of the 
mixed type. Though this is not possible in the canonical 
[444]-color codes with homogeneous edge configurations 
it can be seen to appear in the [444]-color codes with 
either partially or fully mixed edge configurations. We 
show this for the latter case in this section.

Consider a translationally invariant [444]-color code 

shown in Fig. 38. Upon shuffling the \( r_I \) and \( r_{II} \) 

stabilizers we can find the LU between the [444]-color code 

in Fig. 38 and the canonical [444]-color code in Fig. 20. 

Due to the LU we do not expect to see any difference in 
the fundamental topological properties of the canonical 
and the fully mixed [444]-color code of Fig. 38, neverthe- 
less we will now analyze the mixed model in full detail 
to see explicitly how the logical Pauli’s of the mixed type 
appear.
1. Ground State Degeneracy

The GSD or the code space dimension of this model is the same as that of the canonical [444]-color code due to the LU between them. Hence we need to identify the four independent constraints among the stabilizers in Fig. 38.

\[
\beta_1 = \prod_{j=1}^f r_{1j} \prod_{j=1}^f b_{1j}, \ \beta_2 = \prod_{j=1}^f r_{11j} \prod_{j=1}^f g_{11j} \\
\beta_3 = \prod_{j=1}^f g_{1j} \prod_{j=1}^f b_{11j}, \ \beta_4 = \prod_{j=1}^f b_{1j} \prod_{j=1}^f g_{11j} \prod_{j=1}^f g_{11j}.
\]

(155)

2. The Code space

The LU between the homogeneous [444]-color code and the fully mixed case helps us determine the code space. Though we do not write down the ground state of the fully mixed case explicitly we are still interested in identifying the logical Pauli X’s, X_1, X_2, X_3, X_4, to show their mixed nature. While these were relatively easy to construct in the canonical [444]-color code, it requires a bit more thought when the code is made up of mixed edge configurations. We systematically do this by considering the string operators that create the anyonic excitations in this code.

a. Logical Pauli X’s - Any path on a chosen shrunk lattice is made of three types of elementary strings that lie along the three independent directions on the hexagonal lattice as shown in Fig. 39. For the stabilizers in Fig. 38 the elementary strings along the red shrunk lattice are shown in Fig. 40. Extending any of these strings along the indicated directions excites either two or four of the red stabilizers at the end points of the string. The logical Pauli’s corresponding to the red shrunk lattice can now be constructed as shown in Fig. 41. We can follow the same steps to obtain the remaining two logical Pauli X operators by finding the winding operators along the non-contractible loops drawn along the blue shrunk lattice. Consider the elementary strings along the three independent directions on the blue shrunk lattice as shown in Fig. 42. We determine the elementary string operators along the three independent directions on the blue shrunk lattice as shown in Fig. 43 by considering the stabilizers in Fig. 38. Using these elementary string operators we can construct the remaining logical Pauli X’s for the two non-contractible loops along the blue shrunk lattice as shown in Fig. 44. Next we require that the four logical Pauli X’s commute with each other. While this is obvious in the homogeneous [444]-color code, it needs to be checked for the fully mixed case. This is shown diagrammatically in Figs. 45 and 46.

With all logical Pauli X’s in place we can write down all the ground states or encoded qubits of the [444]-color code with mixed edge configurations as

\[
|\bar{a}_1\bar{a}_2\bar{a}_3\bar{a}_4\rangle_{[444]} = \bar{X}_1^{a_1}\bar{X}_2^{a_2}\bar{X}_3^{a_3}\bar{X}_4^{a_4} |\bar{0}_1\bar{0}_2\bar{0}_3\bar{0}_4\rangle_{[444]},
\]

(156)

for \(a_1, a_2, a_3, a_4 \in \{0,1\}\), exhausting the \(2^4\) ground states of the code space for the system on a torus.

b. Logical Pauli Z’s - There are four logical Z’s, two along the red shrunk lattice and two along the blue shrunk lattice as shown in Figs. 47 and 48. Furthermore the logical Z’s commute with each other as shown in Figs. 49 and 50. The logical Z’s measure the global fluxes and can be used to distinguish the states in the code space as

\[
\bar{Z}_1\bar{Z}_2\bar{Z}_3\bar{Z}_4 |\bar{a}_1\bar{a}_2\bar{a}_3\bar{a}_4\rangle_{[444]} = 
\]

\[
|\bar{a}_1\bar{a}_2\bar{a}_3\bar{a}_4\rangle_{[444]}.
\]
c. Relations between the logical X's and Z's - Next we check the commutation and anti-commutation relations between the logical Pauli’s on the code space as in the CSS color codes. The relations between \( \{\bar{X}_1, \bar{X}_2, \bar{X}_3, \bar{X}_4\}, \{\bar{Z}_1, \bar{Z}_2, \bar{Z}_3, \bar{Z}_4\} \) can be summarized as

\[
\{\bar{X}_i, \bar{Z}_j\} = 0, \quad i = j; \quad [\bar{X}_i, \bar{Z}_j] = 0, \quad i \neq j.
\]  
(158)

We check these relations diagrammatically in Figs. 51, 52, 53 and 54.

d. Logicals on the green shrunk lattice - As in the case of the canonical color code, we can construct logical X and Z operators winding along the non-contractible loops drawn along the green shrunk lattice. However we do not expect them to generate new ground states as these operators can be obtained as a product of the logical operators winding along the red and blue shrunk lattices. This is shown in Figs. 55, 56, 57 and 58. We can check that these are indeed the right logicals on the green shrunk lattice by considering the set of elementary string excitations along the three independent directions of the green shrunk lattice as shown in Fig. 59. These considerations imply that it is enough to consider just the red and blue logical operators in analyzing the logical space of this model just as in the original ([336])-color code.

e. Mixed nature of the logical Pauli’s - We claim that all of the logical Pauli operators in this code are of the mixed type, meaning that each of the logical Pauli’s, \( \bar{X}_1, \bar{X}_2, \bar{X}_3, \bar{X}_4 \) and \( \bar{Z}_1, \bar{Z}_2, \bar{Z}_3, \bar{Z}_4 \), comprise of more than one type of Pauli operator. This is evident from their expressions in Figs. 41, 47, 44 and 48. However these are not the only choices for non-contractible loops around a torus. We need to show that the logical operators remain mixed for any choice of non-contractible loop. In other words we need to make sure that the entire equivalence class of logical operators that are related to each other via multiplication by the local stabilizers in Fig. 38 are also mixed.

To see this we note that an arbitrary path winding around the torus is traversed on the hexagonal lattice via three independent directions. And for such an ar-
arbitrary path winding around the torus the logical $X$ or $Z$ is constructed from the elementary red or blue strings shown in Figs. 40 and 43 and since each of these strings along the three different independent directions are mixed we are guaranteed that the entire operator will remain mixed while winding around the torus. The equivalence class of the different logicals are summarized in Fig. 60 The winding operators for different choices of

3. Anyonic content

While the $[444]$-color code shares the same long range features with the original $[336]$-color code, they have different spectrum resulting from single qubit operations as shown in the table in Fig. 62. Most of the excitations resulting from the single qubit operations are immobile, implying that they cannot be moved around the lattice using string operators with zero energy cost. These excitations account for the short-range physics of the model.
In particular they do not contribute to the GSD of the model which is a purely topological feature.

On the other hand the excitations which explain the topological nature of the GSD of the model are the deconfined anyons. For the ground states chosen in Eq. (156) the two sets of $\mathbb{Z}_2$ toric code anyons,

$$\{e, m\}; \{e, m\},$$

are created by the string operators shown in Fig. 63. These string operators can also be obtained by cutting the logical operators such that it no longer winds around the torus.

To ensure that we have indeed obtained the anyons of the $\mathbb{Z}_2 \times \mathbb{Z}_2$ toric code we check their braiding and fusion properties as shown in App. A.

IX. EXAMPLES OF TRIVALENT AND TRICOLORABLE LATTICES OF HIGHER-GENUS TOPOLOGY

In this section we present some examples of trivalent and tricolorable lattices of higher-genus topology with genus two or more.

First let us consider a genus-two surface which can be constructed by preparing two tori, with one hole each, connected by a tube as in Fig. 64. In order to construct trivalent and tricolorable lattices on this topology, we prepare a regular hexagonal lattice of torus topology, remove a face to create a hole, and make two copies of it. Fig. 65 depicts the hexagonal lattices around the holes A and B.

We divide each edge of the holes into three pieces. Then hexagons adjacent to the holes become octagons. Lattice structure of a tube which connects to the tori at
Figure 53. The logical Pauli $X$ and $Z$ operators commute with each other, (a) $[X_1, Z_4] = 0$, $[X_1, Z_3] = 0$, $[X_2, Z_3] = 0$. 

Figure 54. The logical Pauli $X$ and $Z$ operators commute with each other, (a) $[X_3, Z_1] = 0$, $[X_3, Z_2] = 0$, (b) $[X_4, Z_1] = 0$, $[X_4, Z_2] = 0$.

Figure 55. A green logical Pauli $X$ obtained as a product of $\bar{X}_1\bar{X}_3$.

Figure 56. A green logical Pauli $X$ obtained as a product of $\bar{X}_2\bar{X}_4$.

The shortest tube is constructed by squares and hexagons as in the left panel of Fig. 67. For longer tubes, their bulk consists of squares and octagons whereas their boundaries consist of squares and hexagons as in the right panel of Fig. 67.

Connecting either of the tubes to the holed tori at the boundaries A and B, we can construct trivalent and tricolorable lattices realizing genus-two surfaces. We can explicitly confirm the Euler formula for these lattices. Let $k_f$ denote the number of the vertices of the face $f$. Then $3|V| = \sum_f k_f$ holds for any trivalent lattice, where the sum is taken for all the faces of the lattice. Together with this, the formula (14) leads to

$$2 - 2g = \frac{1}{6} \sum_f (6 - k_f).$$

(160)

Since the above lattices consist only of squares, hexagons and octagons, (160) becomes

$$2 - 2g = \frac{1}{3}(|F_4| - |F_8|)$$

(161)

with $|F_4|$ and $|F_8|$ being the number of squares and the number of octagons, respectively. In Fig. 66, we can see that there are six octagons around each of the holes A and B. In the left (right) panel of Fig. 67 there are six squares (24 squares and 18 octagons). From these, we can easily confirm that (161) gives $g = 2$.

Repeating a similar procedure we can construct trivalent and tricolorable lattices realizing surfaces with genus being more than two.

A. Example of a non-CSS color code in higher-genus topology

The trivalent and tricolorable nature of the lattice naturally ensures that the [444]- and [345]- color codes with homogeneous edge configurations are well defined on the higher-genus surfaces. This implies that the red, blue and green faces have the $X$ and $Y$, $Y$ and $Z$, and the $Z$ and $X$ stabilizers for the former case, whereas on the latter code we have $X$ and $Z$, $Y$ and $Z$, and the $Y$ and $Z$ stabilizers, respectively. The other inequivalent classes in the
Figure 57. A green logical Pauli $Z$ obtained as a product of $\bar{Z}_3\bar{Z}_1$.

Figure 58. A green logical Pauli $Z$ obtained as a product of $\bar{Z}_4\bar{Z}_2$.

$[345]$-color codes are obtained by rotations of the colors as seen earlier. For the reason elaborated in Fig. 86 it is not possible to construct a $[345]$-color code where all the vertex-edge combines are confined as the lattice involves square and octagonal faces along with the hexagonal lattice. Nevertheless it is possible to construct $[345]$-color codes with a mixture of confined and deconfined vertex-edge combines, but we will not consider them here.

While the codes with homogeneous edge configurations are rather straightforward to write down, this is not the case where we have either fully mixed or partially mixed edge configurations. We show one such example for the $[444]$-color code with fully mixed edge configurations in Fig. 68. Note that the translational invariance of the $[444]$-color code in Fig. 68 occurs with the periodicity 2. This is a consequence of the way translational invariance for the $[444]$-color code with fully mixed edge configurations is implemented on the the square-octagon lattice (See the fully mixed case in Fig. 89). The hexagonal lattice discretizing the torus is modified as shown in the bottom panel of Fig. 68. We have two sets of stabilizers, those that connect to the red octagon and those that connect to the green octagon. This also elaborates the translational invariance on the torus part. The translationally invariant $[444]$-color code with partially mixed configurations can be similarly written down but we do not show that here. The codes without translational invariance are easier to write down as they are less constraining than the previous case. This is also a result of the algorithm used in the construction of these codes in general.

X. OUTLOOK

We have presented a new approach to generalize the 2D color codes by using the short-range physics of these models. As a result we have discovered new versions, namely the $[444]$ and $[345]$-color codes, that are non-CSS by construction. While these models have the same long-range properties as the color code in its original version, they differ in the short-range physics. Apart from their potential applications in quantum error correction, especially the codes defined on finite lattices like the triangles of different sizes, there are several other interesting directions that one may wish to pursue.

- In this paper we have developed a systematic procedure to construct abelian stabilizer codes on trivalent lattices, with and without boundaries. The
Figure 61. Equivalence of the red and blue logical Pauli $X$ operators. (a) The two winding operators $W_1$ and $W_2$ can be mapped to each other using $g_{IbII}$ between them. (b) In this case they are mapped to each other using $g_{IrIrII}$ between them.

Figure 62. Effect of single qubit operations on the physical qubits of the [444]-color code. See Fig. 5 for the location of the physical qubits on the unit cell comprising the three different colors. Notice that for each of the operations precisely four stabilizers are excited.

construction accommodates several other models, especially those that do not generate an abelian subgroup of the $n$-qubit Pauli group, $\mathbb{P}_n$, once we relax the anticommuting rule. This will then help us to define the model on more general trivalent lattices that include polygons with an odd number of sides.

- This approach for building non-CSS stabilizer codes can be repeated for other kinds of lattices, like the quadrivalent ones that support surface codes like the toric code. However in this case we do not find any new model that is not equivalent to the original surface code by an LU. However this changes when we place two stabilizers on each face of the quadrivalent, bicolorable lattice. In this case we find codes with interesting properties.

- Another direction of generalization appears when we change the number of stabilizers defined on each face. In particular if we include just a single stabilizer on each face we obtain the recently studied honeycomb code [67]. The honeycomb defined in [67] can be thought of as the analog of the [444]-color codes presented in this work. Thus it is natural to write down the [345] analogs of the honeycomb model and explore its properties.

- The toric code can be thought of as a discrete gauge theory in 2D realizing topological order. On the other hand the color code is not a lattice gauge theory in the usual sense, though it still realizes the $\mathbb{Z}_2 \times \mathbb{Z}_2$ topolgical phase. In this context it is interesting to know what other quantum orders can be realized with models like the color code. In this direction the confined [345]-color codes seems to be promising as they have interesting excitations that are constructed using energetic string operators. They seem to resemble a fractonic phase and this direction is worth pursuing as well.

- The codes on higher-genus surfaces are particularly interesting as they enlarge the code space. We have

Figure 63. The string operators creating the fundamental anyons, $\{e, m\}$, of the [444]-color code defined in Fig. 38. The figure shows the operators that have to be applied to the three independent directions of the hexagonal lattice to create anyons at the endpoints of the string.

Figure 64. A genus-two surface constructed by connecting two tori with one hole by a tube. A and B denote the holes of the tori at which the tube is connected.
Figure 65. Example of two copies of a hexagonal lattice of tori with holes A and B. For each of the two, the upper and lower horizontal dashed lines are identified, and the left and right vertical dashed lines are also.

Figure 66. Each edge of the holes in Fig. 65 is divided into three pieces. Then hexagons adjacent to the holes become octagons. Colors around A and B indicate the colors of the boundaries of the tube.

Figure 67. The shortest tube (the left panel) and a longer tube (the right panel). In each figure, the upper and lower horizontal dashed lines are identified.

showed just a single model in this setting. There are more possibilities in this direction which we hope to visit in future publications.

- It would also be interesting to find appropriate generalizations of this construction on higher-dimensional lattices and hyperbolic spaces.

- In Apps. F2, F1, F3 we have illustrated stabilizer codes with mixed vertex energetics. It would be interesting to explore their properties as well, though we expect the long range physics to coincide with that of the $Z_2 \times Z_2$ topological phase.

- Finally in Apps. G, H we give a glimpse of a code which does not satisfy the anticommuting rule. These models have quasi-local symmetries which warrants further study.

ACKNOWLEDGMENTS

PP and AC would like to thank IIT-BBS where this work was carried out. AC acknowledges support from IIT-BBS seed grant (No. SP103). We thank Michael Vasmer and Paulo Teotonio-Sobrinho for useful discussions and references.

Appendix A: Fusion and braiding rules for [444]-color codes with fully mixed edge configurations

Here we discuss the braiding and fusion rules of the anyons in the fully mixed [444]-color codes.

a. Braiding rules - It is easy to see that the electric charges and magnetic fluxes of the same color do not pick up a phase when they move around each other. This is due to the fact that two strings of the same color never cross each other at a vertex or the location of a physical qubit.

Thus we should check the braiding of particles of different color around each other. We expect to see a phase of -1 when $e$ goes around $m$ and when $e$ goes around $m$. However as the anyons can be oriented in three different directions we need to check that the -1 phase is obtained in all cases. This is verified in Figs. 69 and 70. Furthermore we need to ensure that $e$ and $e$, and $m$ and $m$ are mutually bosonic. This is shown in Figs. 71 and 72.

b. Fusion rules - The anyons $\{e, m\}$ obey the fusion rules of the $Z_2 \times Z_2$ toric code,

$$e \times e = e \times e = e = m \times m = m \times m = 1,$$

$$e \times m = e m, \quad e \times m = e m,$$

$$e \times m = e m, \quad e \times m = e m.$$  

(A1)
Figure 68. A translationally invariant [444]-color code with fully mixed edge configurations on a higher-genus surface. The top panel shows the code on the tube connecting two tori and on octagons adjacent to the hole of each torus. The bottom panel shows the code defined on a hexagonal lattice on a holed torus. The left end of the upper left (right) figure is identified with the right end of the lower left (right) figure.

along with $e \times e$ and $m \times m$ which correspond to direct products of its components. There are four kinds of ‘dyonic’ excitations in this case, $\epsilon_{em}, \epsilon_{em}, \epsilon_{em}, \epsilon_{em}$. In a diagrammatic way (See Fig. 73) we ensure that we obtain the vacuum in case two particles of the same type fuse and that dyonic charges are obtained when particles of the same color fuse.

Appendix B: Non-translationally invariant color code models

Here we write down the non-translationally invariant non-CSS color codes on two dimensional trivalent and tricolorable lattices. We emphasize that the source for the loss of translational invariance is not the lattice geometry, i.e. we will continue to work on either the hexagonal or the square-octagonal lattice, but now the stabilizer pairs, on each face of a chosen color, are different as we move along the shrunk lattice of the same color.

1. Non-translationally invariant [444]-color codes

The non-translationally invariant [444]-color codes on the hexagonal lattice are shown in Fig. 74. Notice that is not possible to give up translational invariance and have homogeneous edge configurations.
2. Non-translationally invariant and deconfined [345]-color codes

The non-translationally invariant [345]-color codes with partially and fully mixed edge configurations corresponding to the three inequivalent classes are shown in Figs. 75 and 76, for the hexagonal lattice.

The LU to the canonical deconfined [345]-color codes are shown in Fig. 77.

3. Non-translationally invariant and confined [345]-color codes

The consistent models on the hexagonal lattice, with non- translational invariance, are shown in Figs. 78 and 79.
4. Non-translationally invariant \([336]\)-color codes

The non-CSS \([336]\)-color codes on the hexagonal lattice for partially and fully mixed edge configurations, without translational invariance is shown in Fig. 80.

Appendix C: Steps for local code generation for deconfined and confined \([345]\)-color codes and the \([336]\)-color codes

The steps for the local construction in the deconfined \([345]\) setting are illustrated in Fig. 81.

Fig. 82 shows the steps for the local construction of the confined \([345]\)-color codes.

The local construction for the \([336]\)-color code is shown in Fig. 83.

Appendix D: The other inequivalent classes of the deconfined and confined \([345]\)-color codes with translational invariance

There are three inequivalent classes for the bulk vertices of the \([345]\) type (See Fig. 12). The class \(I\) models for both the deconfined and confined varieties with all possible edge configurations were shown in the main text. Here we complete the classification by writing down the class \(II\) and \(III\) \([345]\)-color codes in all its variations.
Figure 77. The LU’s between the canonical deconfined [345]-color code on the hexagon lattice and the (a) partially mixed, (b) fully mixed non-translationally invariant deconfined [345]-color codes.

The translationally invariant [345]-color codes with partially and fully mixed edge configurations corresponding to classes II and III are shown in Figs. 84 and 85, for the hexagonal lattice.

Fig. 86 explores the polygons where the confined [345]-color codes can be defined consistently.

The class II and III confined [345]-color codes are shown in Figs. 87 and 88.

Figure 78. The confined [345]-color codes on the hexagonal lattice with partially mixed edge configurations and without translational invariance. This code has vertices belonging to the three inequivalent [345] bulk vertices (See Fig. 12).

Appendix E: Non-CSS color codes on square-octagon lattices on a torus

1. [444]-color codes on the (4, 8, 8) lattice

The translationally invariant models are shown in Fig. 89, that are LU related to each other. We can also write down a model without translational invariance on the square-octagon lattice where the edges have partially (fully) mixed configurations, in Fig. 90.
2. Deconfined [345]-color codes

The deconfined [345]-color codes with and without translational invariance on the (4,8,8)-lattice are shown in Figs. 91, 92, 93, 94 and 95.

Appendix F: Non-CSS color codes with mixed bulk vertex types

In this appendix we present further possibilities that open up with the algorithm for constructing two-dimensional color codes introduced in this paper. We focus our attention on color codes which have more than one type of vertex, namely a mixture of [345] and [444], a mixture of [345] and [336] or a mixture of all three types of vertices. It is clear that by construction these models are not LU equivalent to each other or the color codes in the main text which sport just a single vertex type. There are plenty of possibilities as a result, and we show just a few of them relegating a more thorough analysis to future papers.

1. [345] − [444]-color codes

Here we consider color codes that have vertices of both the [345] and the [444] types. We show two of the simplest such examples on hexagonal lattices in Fig. 96 where the vertices of a single hexagon is of a different type compared to the rest of the vertices. In this setup the two possibilities are denoted [345] − ([444])- and [444] − ([345])-color codes where the defect hexagons contain the vertex types indicated in the () . We can consider more general color codes with an increased number of energetic defects such that they are placed in a random manner on the hexagonal lattice.
Figure 81. The steps for the local construction of a deconfined [345]-color code. The procedure can be continued to extend the code on a finite trivalent lattice just as in the case of the [444]-color code. In step 1 we have chosen the first class of the three inequivalent choices for a [345] Y vertex (See Fig. 12). While choosing the edge configurations in the second step we can go with either a (a) homogeneous (no mixing), (b) partially mixed or (c) fully mixed edge configuration. In the third step we choose the second vertex to be of the [345] type and such that the vertex-edge combine leads to a deconfined stabilizer pair. In step 4 the procedure is continued for another edge where we continue to maintain the deconfined nature of the [345]-color code. We show just one among the many possibilities in this step.
2. \([345] - [336]\)-color codes

The color codes here include vertices of types \([345]\) and \([336]\). The simplest examples where there is a single \([345]\) type hexagon in a sea of \([336]\) vertices or vice versa are shown in Fig. 97.

3. \([444] - [345] - [336]\)-color codes

Finally we consider color codes where all the three bulk vertex types are present. In the examples shown in Fig. 98, we pick the vertices on two of the hexagons to be of different vertex types with respect to the remaining vertices.

Appendix G: \([255]\)-color codes

Here we briefly look at other possibilities that emerge from possible Y-vertex configurations listed in Table II, particularly the \([255]\) vertex type. These energy values are a result of the configurations containing, \{\textit{xyzzzz}\} and their LU equivalents. Stabilizer pairs created using these bulk vertices do not obey the anticommutation rule stated in Sec. IV. As a consequence the color codes made up of these vertices have just a single type of stabilizer on one of the colored faces. In Fig. 99 we show translationally invariant \([255]\)-color codes on the hexagonal lattice with homogeneous, partially and fully mixed edge configurations. Notice that while the blue and green faces have two stabilizers each, as it happens in the \([336]\)-, \([345]\)- and \([444]\)-color codes, the red faces have just a single stabilizer. A crucial difference between these models and the \([336]\)-, \([444]\)- or \([345]\)-color codes is that these models are not entirely topological. This is seen due to the presence of weight-two symmetries supported on the edges connecting red faces. \(z_{v_1}z_{v_2}\) is a symmetry for the edge containing the vertices, \(v_1\) and \(v_2\) (See Fig. 99 (a)). Moreover there are a fewer number of independent constraints among the stabilizers in this case,

\[
C_1 = \prod_{j=1}^{f} r_j \prod_{j=1}^{f} b_{11j} = 1, \quad C_2 = \prod_{j=1}^{f} r_j \prod_{j=1}^{f} g_{11j} = 1, \\
C_3 = \prod_{j=1}^{f} g_{11j} \prod_{j=1}^{f} b_{11j} \prod_{j=1}^{f} b_{11j} = 1, \quad (G1)
\]

three as opposed to four found for the \([336]\)-, \([444]\)- and \([345]\)-color codes. Thus the number of independent stabilizers on the hexagonal lattice becomes, \(v - \frac{|F|}{4} + 3 = \frac{|F|}{4} + 3\), making the code space dimension \(2^{\frac{|F|}{4} + 3}\), an extensive quantity.

Appendix H: \([345] - ([255])\)-color codes

Fig. 100 shows a translationally invariant deconfined \([345]\)-color code with a ‘defect’ hexagon made of \([255]\) vertices. Note that there is just one stabilizer on the defect hexagon. Any string operator made of the Pauli \(x\)’s along the red shrunk lattice connecting these two defects is a symmetry of this code. We can think of these as a quasi-local symmetry.
Figure 82. The steps to construct the stabilizer pair of the confined [345]-color code. A [345] bulk vertex of type 1 (See Fig. 12) has been chosen in step 1. The construction proceeds in a similar manner for the other two inequivalent [345] bulk vertices. The possible edge configurations for the stabilizer pair are shown in step 2 with (a) homogeneous (no mixing), (b) partially and (c) fully mixed edge configurations. In step 3 the second vertex is chosen such that the vertex is of the [345] type and the vertex-edge combine is of the confined type. In step 4 the procedure is extended to one more edge. There are more choices possible in step 4 though we just show one of them.
Figure 83. The steps in the construction of the [336]-color codes on a local part of the trivalent lattice. The choice of the bulk [336] vertex is equivalent to the original color code by an LU. The edge configurations are shown in Step 2 - (a) homogeneous, (b) partially and (c) fully mixed edge configurations. The choice for the bulk vertex in Step 3 is determined by the choice of the edge configurations in Step 2. The construction is continued for another edge in Step 4. The resulting codes are equivalent to the original [336]-color code by an LU.
Figure 84. Translationally invariant, deconfined [345]-color codes with (a) homogeneous, (b) partially and (c) fully mixed edge configurations.

Figure 85. Translationally invariant, deconfined [345]-color codes with (a) homogeneous, (b) partially and (c) fully mixed edge configurations.
Figure 86. Polygons where the confined [345]-color code can be consistently defined are shown in, (a) a square and (b) a hexagon. On (c) the octagon the edges indicated in orange form deconfined vertex-edge combines and in (d) the decagon, the orange dot indicates a [444] bulk vertex.

Figure 87. The Class II confined [345]-color codes on the hexagonal lattice with (a) homogeneous, (b) partially and (c) fully mixed edge configurations.
Figure 88. The Class III confined [345]-color codes on the hexagonal lattice with (a) homogeneous, (b) partially and (c) fully mixed edge configurations.

Figure 89. Translationally invariant [444]-color codes with (a) homogeneous, (b) partially and (c) fully mixed edge configurations on the square-octagon lattice.
Figure 90. Non-translationally invariant [444]-color codes with (a) partially mixed and (b) fully mixed edge configurations.

Figure 91. Translationally invariant, deconfined [345]-color codes on the square-octagon lattice with homogeneous edge configurations. The three inequivalent classes can be distinguished by the bulk vertex configurations at the orange dot.
Figure 92. Translationally invariant, deconfined [345]-color codes on the square-octagon lattice with partially mixed edge configurations. The three inequivalent classes can be distinguished by the bulk vertex configurations at the orange dot. Notice that the red stabilizers in this code are translationally invariant with the period 2.

Figure 93. Translationally invariant, deconfined [345]-color codes on the square-octagon lattice with fully mixed edge configurations. The three inequivalent classes are distinguished by the bulk vertex configurations at the orange dot. Notice that the red stabilizers in this code are translationally invariant with the period 2.
Figure 94. Non-translationally invariant, deconfined [345]-color codes on the square-octagon lattice with partially mixed edge configurations.

Figure 95. Non-translationally invariant, deconfined [345]-color codes on the square-octagon lattice with fully mixed edge configurations.
Figure 96. Example of color codes with mixed vertex types. (a) The [345] − ([444])-color code. All the vertices are of the [345] type except on the hexagon with orange vertices which are of the [444] type. (b) The [444] − ([345])-color code. Here the situation is reversed with all the vertices of the [444] type except the orange ones which are of the [345] type. In the former we can interpret the code as including an energetic defect of the [444] type in the [345]-color code, whereas in the latter it is the opposite. These models are translationally invariant, with the stabilizer set $II$, containing $z$ Pauli operators on all the blue faces except the on the defect indicated with orange dots.

Figure 97. Example of color codes with mixed vertex types. (a) The [336] − ([345])-color code. All the vertices are of the [336] type except on the hexagon with orange vertices which are of the [345] type. (b) The [345] − ([336])-color code. Here the situation is reversed with all the vertices of the [345] type except the orange ones which are of the [336] type. In the former we can interpret the code as including an energetic defect of the [345] type in the [336]-color code, whereas in the latter it is the opposite. These models are translationally invariant, with the stabilizer set $II$, containing (a) $z$ Pauli operators on all the blue faces except the on the defect indicated with orange dots and the stabilizer set $I$, containing (b) $x$ Pauli on all the green faces except on the one indicated by the orange dots.
Figure 98. Example of color codes containing all three vertex types. (a) The $[345] - ([336] - [444])$-color code. All the vertices are of the $[345]$ type except the vertices indicated by orange and purple dots which are of the $[444]$ and $[336]$ types respectively. (b) The $[336] - ([444] - [345])$-color code. Here all the vertices of the $[336]$ type except the orange and purple ones which are of the $[345]$ and $[444]$ types respectively. (c) The $[444] - ([345] - [336])$-color code. Here all the vertices of the $[444]$ type except the orange and purple ones which are of the $[345]$ and $[336]$ types respectively. In this case each code has two energetic defects, placed next to each other. All these codes are translationally invariant, with the stabilizers modified at the location of the two hexagon defects indicated by the orange and the purple dots.

Figure 99. Translationally invariant $[255]$-color codes on the hexagonal lattice. Figure shows the different possible edge configurations, (a) homogeneous, (b) partially mixed and (c) fully mixed. Note that there is just one type of stabilizer on each of the red faces for the three codes. It is easy to verify that these models are equivalent to each other by LU’s.
Figure 100. Translationally invariant $[345]$-color code on the hexagonal lattice with a $[255]$ ‘defect’. The vertex-edge combine around the $[255]$-defect hexagon are of the confined type.
A. Kubica, The abcs of the color code: A study of topological quantum codes as toy models for fault-tolerant quantum computation and quantum phases of matter (2018).

M. Vasmer and D. E. Browne, Three-dimensional surface codes: Transversal gates and fault-tolerant architectures, Physical Review A (2019).

T. R. Scruby and D. E. Browne, A hierarchy of anyon models realised by twists in stacked surface codes, Quantum 4, 251 (2020).

B. Yoshida, Classification of quantum phases and topology of logical operators in an exactly solved model of quantum codes, Annals of Physics 326, 15 (2011).

F. Pastawski, B. Yoshida, D. Harlow, and J. Preskill, Holographic quantum error-correcting codes: toy models for the bulk/boundary correspondence, Journal of High Energy Physics 2015, 1 (2015).

A. Kubica and B. Yoshida, Ungauging quantum logical operators in an exactly solved model of qudit color codes and toric codes, Physical Review A 100, 012348 (2019).

B. Yoshida and I. L. Chuang, Framework for classifying logical operators in stabilizer codes (2010).

H. Bombin, Dimensional jump in quantum error correction, arXiv: Quantum Physics (2014).

H. Bombin, An introduction to topological quantum codes, arXiv: Quantum Physics (2013).

A. Kubica, B. Yoshida, and F. Pastawski, Unfolding the color code, New Journal of Physics 17, 083026 (2015).

P. K. Sarvepalli and K. R. Brown, Topological subsystem codes from graphs and hypergraphs, Physical Review A 86, 042336 (2012).

P. K. Sarvepalli and R. Raussendorf, Efficient decoding of topological color codes, Physical Review A 85 (2012).

A. B. Aloshious and P. K. Sarvepalli, Decoding toric codes on three dimensional simplical complexes, IEEE Transactions on Information Theory 67, 931 (2021).

B. J. Brown, N. Nickerson, and D. E. Browne, Fault tolerance with the gauge color code, arXiv: Quantum Physics (2015).

H. G. Katzgraber, H. Bombin, and M. A. Martin-Delgado, Error threshold for color codes and random three-body ising models., Physical review letters 103 9, 090501 (2009).

M. Beverland, O. Buerschaper, R. Koenig, F. Pastawski, J. Preskill, and S. Sijher, Protected gates for topological quantum field theories, arXiv: Quantum Physics (2014).

J. W. Harrington, Analysis of quantum error-correcting codes: symplectic lattice codes and toric codes (2004).

C. Horsman, A. G. Fowler, S. J. Devitt, and R. V. Meter, Surface code quantum computing by lattice surgery, New Journal of Physics 14, 123011 (2012).

R. S. Andrist, Understanding topological quantum error-correcting codes using classical spin models (2012).

A. Kubica and B. Yoshida, Unfolding the topological color codes, Physical Review A (2018).

A. B. Aloshious and P. K. Sarvepalli, Projecting three-dimensional color codes onto three-dimensional toric codes, Phys. Rev. A 98, 012302 (2018).

A. B. Aloshious and P. K. Sarvepalli, Local equivalence of qudit color codes and toric codes, Physical Review A 100, 012348 (2019).

A. Kubica and B. Yoshida, Unfolding quantum error-correcting codes, arXiv preprint arXiv:1805.01836 (2018).

A. Kubica, B. Yoshida, and F. Pastawski, Unfolding the color code, New Journal of Physics 17, 083026 (2015).

M. Vasmer and A. Kubica, Morphing quantum codes (2021).

A. M. Stephens, W. J. Munro, and K. Nemoto, High-threshold topological quantum error correction against biased noise, Physical Review A 88, 060301 (2013).

D. K. Tuckett, S. D. Bartlett, and S. T. Flammia, Ultrahigh error threshold for surface codes with biased noise, Physical review letters 120, 050505 (2018).

D. K. Tuckett, A. S. Darmawan, C. T. Chubb, S. Bravyi, S. D. Bartlett, and S. T. Flammia, Tailoring surface codes for highly biased noise, Physical Review X 9, 041031 (2019).

D. K. Tuckett, S. D. Bartlett, S. T. Flammia, and B. J. Brown, Fault-tolerant thresholds for the surface code in excess of 5% under biased noise, Physical review letters 124, 130501 (2020).

The constraint relations quoted in the literature read as \( \prod_j X_{f_j} = \prod_j X_{f_j} = \prod_j X_{f_j} \) and \( \prod_j Z_{f_j} = \prod_j Z_{f_j} = \prod_j Z_{f_j} \). This is easily seen to be equivalent to the constraint relations used in Eq. (12).

M. B. Hastings and J. Haah, Dynamically generated logical qubits, Quantum (2021).