SciPost Phys. 16, 033 (2024)

Error-correcting codes for fermionic quantum simulation

Yu-An Chen[1⋆], Alexey V. Gorshkov[2] and Yijia Xu[2,3†]

1 International Center for Quantum Materials, School of Physics, Peking University, Beijing 100871, China
2 Joint Quantum Institute and Joint Center for Quantum Information and Computer Science, NIST/University of Maryland, College Park, Maryland 20742, USA
3 Institute for Physical Science and Technology, University of Maryland, College Park, Maryland 20742, USA

⋆ yuanchen@pku.edu.cn , † yijia@umd.edu

Abstract

Utilizing the framework of $\mathbb{Z}_2$ lattice gauge theories in the context of Pauli stabilizer codes, we present methodologies for simulating fermions via qubit systems on a two-dimensional square lattice. We investigate the symplectic automorphisms of the Pauli module over the Laurent polynomial ring. This enables us to systematically increase the code distances of stabilizer codes while fixing the rate between encoded logical fermions and physical qubits. We identify a family of stabilizer codes suitable for fermion simulation, achieving code distances of $d = 2, 3, 4, 5, 6, 7$, allowing correction of any $\lfloor \frac{d-1}{2} \rfloor$-qubit error. In contrast to the traditional code concatenation approach, our method can increase the code distances without decreasing the (fermionic) code rate. In particular, we explicitly show all stabilizers and logical operators for codes with code distances of $d = 3, 4, 5$. We provide syndromes for all Pauli errors and invent a syndrome-matching algorithm to compute code distances numerically.

Received 16-08-2023
Accepted 09-01-2024
Published 26-01-2024
doi:10.21468/SciPostPhys.16.1.033

Contents

1 Introduction

Error-correcting codes were initially developed to correct quantum errors on noisy quantum devices and have found further applications in condensed matter physics and high-energy physics. The cornerstone of quantum error correction is the stabilizer formalism [1], which defines the codewords in the common eigenspace of elements in an Abelian group, referred to as the stabilizer group. A stabilizer code is labeled $[[n, k, d]]$ when it uses n physical qubits to encode k logical qubits with code distance d. The code distance is the minimum weight of an operator that commutes with all elements in the stabilizer group but is not in the stabilizer group itself. The ratios $\frac{k}{n}$ and $\frac{d}{n}$ determine the quality of codes. Recent developments show the existence of "good" quantum low-density parity-check (LDPC) codes, i.e., with the number of logical qubits k and the code distance d both scaling linearly with the number of physical qubits n [2–5]. In this paper, our focus is on a different objective. Instead of encoding logical qubits, we aim to encode logical fermions using physical qubits. This motivation comes from the need to simulate fermions on quantum computers since most models of matter involve electrons, which are, in fact, fermions [6–14]. While fault-tolerant quantum computation [15, 16] is the ultimate goal, current devices still have limited resources and suffer from noise, so error-mitigation schemes are crucial. Therefore, we seek an effective design such that when we implement fermions with qubits on a quantum computer, certain physical qubit errors can be corrected directly in this protocol without having to encode the underlying qubits further. Thus, we want to systematically increase the code distance d in a fermion-to-qubit mapping with a fixed code rate (between logical fermions and physical qubits).[1]

When a fermionic Hamiltonian consists of geometrically local terms, they can be mapped to local qubit operators by the Bravyi-Kitaev superfast encoding and its variants [8, 17], by the auxiliary methods [18–23], or by exact bosonization [24–26]. The mappings between fermionic Hamiltonians and higher-spin Hamiltonians are also studied in Refs. [27–30]. There are also proposals that utilize defects of surface codes for fermionic quantum simulation [31–33], and those defects are recently implemented by Google Quantum AI [34]. Variants of these mappings have been studied to optimize different costs [35–48]. In the context of quantum many-body physics, these mappings also reveal the deep connections between fermion and spin systems [49, 50]. There is another exact fermion-flux lattice duality derived from the $\mathbb{Z}_2$ gauge theory [51]. Aside from the investigation of constructing new mappings, fermion-to-qubit mappings have been studied in the context of variational quantum circuits [52, 53]. In all the above-mentioned methods, extra qubits are required, e.g., the number of qubits is twice the number of fermions on a 2d square lattice. This is the price for the locality-preserving

[1] The code rate here is defined as the ratio between the number of logical fermionic modes k and the number of physical qubits n, in the $n \to \infty$ limit. Each code will be demonstrated in an infinite plane, but they can be defined on a torus or open disk (up to some boundary modifications) with linear size L, such that both n and k scale with L^2. If L is sufficiently large, the boundary effects are negligible, and the ratio k/n will converge to the code rate.

property.[2] These methods can be thought of as stabilizer codes. Given N fermions with the Hilbert space dimension 2^N, they are mapped to $2N$ qubits with space dimension 2^{2N}, which is an enlarged space. After N gauge constraints (stabilizer conditions) are imposed, the gauge-invariant subspace (code space) has dimension 2^N, which matches the dimension of the logical fermions. It has been shown that gauge constraints can be utilized for error correction [54], and code distances can be studied for these stabilizer codes. Ref. [17] demonstrates that an improved Bravyi-Kitaev superfast encoding can correct any single-qubit error in a graph where each vertex has degree $d \geq 6$. In Ref. [55], another version of the Bravyi-Kitaev superfast encoding is proposed, called the "Majorana loop stabilizer code," which is designed to have code distance $d = 3$ such that any single-qubit error can be corrected. However, it is not known how to generalize the Bravyi-Kitaev superfast encoding to produce codes with higher and higher code distances. An alternative approach is code concatenation, where logical qubits in error-correcting codes replace physical qubits in the fermion-to-qubit mappings. However, code concatenation will decrease the code rate between logical fermions and physical qubits, increasing the overhead of fermionic simulation. In this work, we present a method that increases the code distances of the fermion-to-qubit mappings while preserving the code rate.

In this paper, we conjugate an existing stabilizer code with a Clifford circuit.[3] This produces a new stabilizer code. Since the new code is obtained via conjugation by a unitary operator, the algebra of the logical operators is preserved. If we choose the circuit wisely, the new stabilizer code will have a larger code distance ($d \geq 3$), compared to the code distance $d = 2$ of the original exact bosonization. To study Clifford circuits systematically, we utilize the Laurent polynomial method introduced in Refs. [56, 57] and further extended in Ref. [58], which shows that any Pauli operator can be written as a vector in a symplectic space. For a system with translational symmetry, e.g., the 2d square lattice, the space of Pauli operators becomes a module over a polynomial ring. Furthermore, this polynomial method can be used to formulate the 2d bosonization concisely [59]. The commutation relations of Pauli operators are determined by the symplectic form. Ref. [60] shows that there is a one-to-one correspondence (up to a translation operator on the lattice):

$$\text{Automorphism of the symplectic form} \iff \text{Clifford circuit on a 2d square lattice.}$$

Therefore, the problem of finding new codes turns into a problem of searching for "good"[4] automorphisms of the Pauli module with the symplectic form, which can be achieved efficiently by exhaustive numerical search.

In this work, we use the Laurent polynomial method to construct bosonizations on a 2d square lattice. Table 1 is the summary of our results. In Section 2, we review the original 2d bosonization method [24] in Section 2.1 and then pictorially construct 2d bosonizations with distances of 3, 4, and 5 in Section 2.2, 2.3, and 2.4, respectively. We review the Laurent polynomial method in Section. 3.1. In Section. 3.2, we describe all these bosonizations within the framework of the Laurent polynomial method. In addition, in Section. 3.3, we describe a computerized method to search for bosonizations. In Appendix A, we discuss the "syndrome matching" method used to compute the code distance of a given bosonization. In Appendix B, we describe the generators of the symplectic group and choose sixteen elementary automorphisms for our numerical search algorithm. In Appendix C, we show the polynomial representations of an automorphism with a distance of 6 and another with a distance of 7.

[2] We can apply the Jordan-Wigner transformation on the 2d lattice by choosing a path including all vertices. However, some local fermionic terms will be mapped to long string operators that are highly nonlocal.

[3] A circuit U is Clifford if and only if $UPU^\dagger$ is a product of Pauli matrices for any given Pauli matrix P.

[4] Here, "good" refers to symplectic automorphisms that generate bosonizations with higher code distances while preserving locality and code rates.

Table 1: A comparison of codes based on modified exact bosonization to the Bravyi-Kitaev superfast encoding [8] and to the Majorana loop stabilizer code [55] on a 2d square lattice. The $d = 2$ exact bosonization is equivalent to the Bravyi-Kitaev superfast encoding with a specific choice of the ordering of edges [48]. We list the code distance, as well as the weights (after mapping to qubits) of a fermion occupation term (local fermion parity term), of a hopping term, and of a density-density interaction between nearest neighbors. The weights of the stabilizers are also shown.

	distance	occupation	hopping	interaction	stabilizer
Bravyi-Kitaev superfast encoding	2	4	6	6	6
Majorana loop stabilizer code	3	3	3-4	4-6	4-10
Exact bosonization ($d = 3$)	3	4	3-5	6	8
Exact bosonization ($d = 4$)	4	6	5-6	10	10
Exact bosonization ($d = 5$)	5	8	5-9	12-14	12
Exact bosonization ($d = 6$)	6	12	6-13	16-20	18
Exact bosonization ($d = 7$)	7	12	7-23	16-18	26

2 Results

In Section 2.1, we begin by reviewing the original 2d bosonization on a square lattice from Ref. [24]. Then we demonstrate a new way to perform bosonization with code distances of $d = 3$, $d = 4$, and $d = 5$ in Section 2.2, Section 2.3, and Section 2.4, respectively.

2.1 Review of the original bosonization

We first describe the Hilbert space in Fig. 1. The elements associated with vertices, edges, and faces will be denoted by v, e, and f, respectively. On each face f of the lattice, we place a single pair of fermionic creation-annihilation operators $c_f, c_f^\dagger$, or equivalently a pair of Majorana fermions γ_f, γ_f'. The even fermionic algebra consists of local observables with trivial fermionic parity, i.e., local observables that commute with the total fermion parity $(-1)^F \equiv \prod_f (-1)^{c_f^\dagger c_f}$ where $F = \sum_f c_f^\dagger c_f$ is the total fermion number.[5] The even algebra is generated by [24]:

[5] The even fermionic algebra can also be considered as the algebra of local observables containing an even number of Majorana operators.

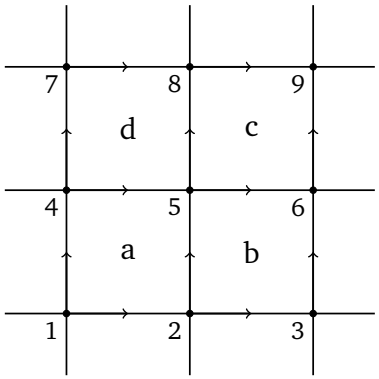


Figure 1: Bosonization on a square lattice [24]. We put Pauli matrices X_e, Y_e, and Z_e on each edge and one complex fermion $c_f, c_f^\dagger$ on each face. We will work in the Majorana basis $\gamma_f = c_f + c_f^\dagger$ and $\gamma_f' = -i(c_f - c_f^\dagger)$ for convenience.

1. On-site fermion parity (occupation):

$$P_f \equiv -i\gamma_f\gamma'_f\,.^{6} \tag{1}$$

2. Fermionic hopping term:

$$S_e \equiv i\gamma_{L(e)}\gamma'_{R(e)}\,, \tag{2}$$

where $L(e)$ and $R(e)$ are faces to the left and right of e, with respect to the orientation of e in Fig. 1.

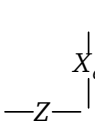
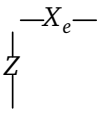
The bosonic dual of this system involves $\mathbb{Z}_2$-valued spins on the edges of the square lattice. For every edge e, we define a unitary operator U_e that squares to 1. Labeling the faces and vertices as in Fig. 1, we define:

$$\begin{aligned} U_{56} &= X_{56}Z_{25}\,, \\ U_{58} &= X_{58}Z_{45}\,, \end{aligned} \tag{3}$$

where X_e, Z_e are Pauli matrices acting on a spin at edge e:

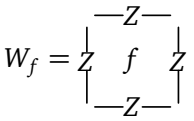
$$X_e = \begin{bmatrix} 0 & 1 \\ 1 & 0 \end{bmatrix}, \quad Z_e = \begin{bmatrix} 1 & 0 \\ 0 & -1 \end{bmatrix}. \tag{4}$$

Operators U_e for the other edges are defined by using translation symmetry. Pictorially, operator U_e is depicted as

$U_e =$  or  (5)

corresponding to the vertical or horizontal edge e.

In Ref. [24], U_e and S_e are shown to satisfy the same commutation relations. We also map the fermion parity P_f at each face f to the "flux operator" $W_f \equiv \prod_{e\subset f} Z_e$, the product of Z_e around a face f:

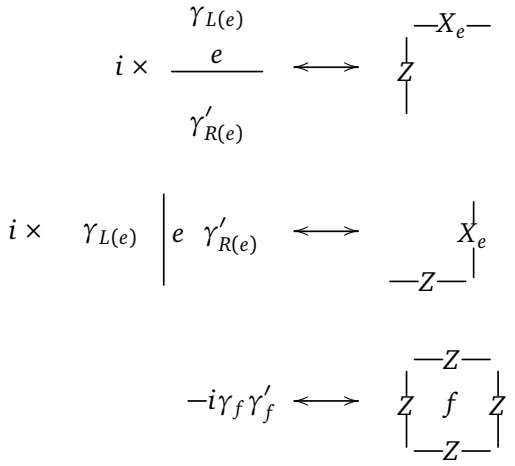
$W_f =$  (6)

The bosonization map is

$$\begin{aligned} S_e &\longleftrightarrow U_e\,, \\ P_f &\longleftrightarrow W_f\,, \end{aligned} \tag{7}$$

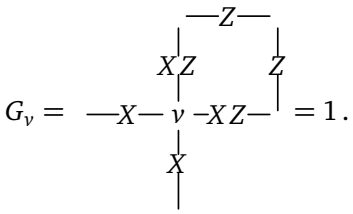
[6] $P_f = -i\gamma_f\gamma'_f = (-1)^{c_f^\dagger c_f}$ measures the occupancy of the fermionic mode at face f.

or pictorially



(8)

The condition $P_a P_c S_{58} S_{56} S_{25} S_{45} = 1$ on fermionic operators gives a gauge (stabilizer) constraint $G_v = W_{f_c} \prod_{e \supset v_5} X_e = 1$ for bosonic operators, or generally



(9)

The gauge constraint Eq. (9) can be considered as the stabilizer ($G_v|\Psi\rangle = |\Psi\rangle$ for $|\Psi\rangle$ in the code space), which forms the stabilizer group $\mathcal{G}$. The operators U_e and W_f generate all logical operators.[7] The weight of a Pauli string operator O is the number of Pauli matrices in O, denoted as $\mathrm{wt}(O)$. For example, we have $\mathrm{wt}(U_{56}) = \mathrm{wt}(U_{58}) = 2$, $\mathrm{wt}(W_f) = 4$, and $\mathrm{wt}(G_v) = 6$. The code distance d is defined as the minimum weight of a logical operator excluding stabilizers:

$$d = \min\{\mathrm{wt}(O) \mid [O, \mathcal{G}] = 0,\ O \notin \mathcal{G}\}\,. \tag{10}$$

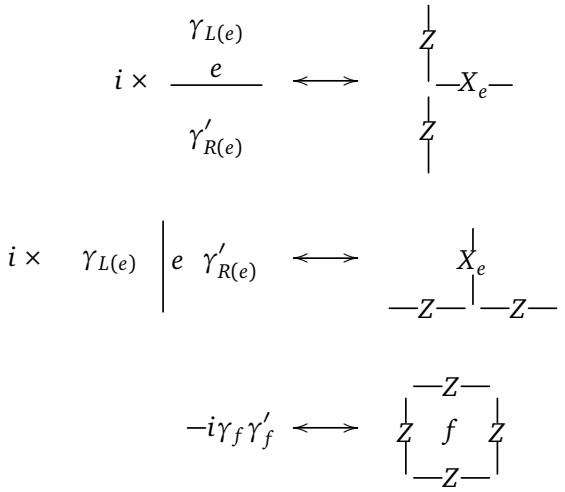
The code distance of this original bosonization is $d = 2$ as U_e has weight 2 and any single Pauli matrix violates at least one G_v, which implies that there is no logical operator with weight 1.

There are four types of nearest-neighbor hopping terms ($\gamma_L\gamma'_R$, $\gamma_L\gamma_R$, $\gamma'_L\gamma'_R$, and $\gamma'_L\gamma_R$) and one type of fermion occupation term ($-i\gamma_f\gamma'_f$). When mapped to Pauli matrices, their weights wt_i are in the range $2 \le \mathrm{wt}_i \le 6$. The maximum weight corresponds to the worst case to simulate the fermion hopping term or the fermion occupation term. A good stabilizer code requires a balance between the minimum weight and the maximum weight. A high minimum weight guarantees the error-correcting property, while a low maximum weight implies that the cost of simulation is low. We label the minimum and maximum weights of the hopping terms as $\mathrm{wt}_{\min}$ and $\mathrm{wt}_{\max}$. In this example, $(\mathrm{wt}_{\min}, \mathrm{wt}_{\max}) = (2, 6)$.

2.2 Bosonization with code distance $d = 3$

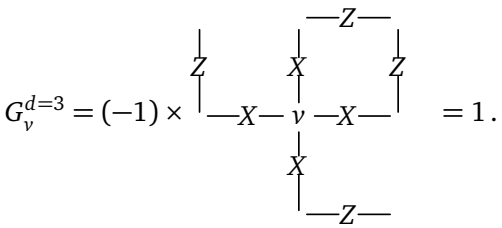
We now introduce a new way to map the fermionic operators S_e and P_f to Pauli matrices. For simplicity, we present the mapping in a pictorial way:

[7] The logical operators consist of all operators that commute with $\mathcal{G}$. Elements of $\mathcal{G}$ are trivial logical operators as stabilizers have no effect on the code space. U_e and W_f together generate all the other logical operators.



(11)

The stabilizer on the bosonic side is



(12)

Notice that there is a minus sign coming from $ZXZ = -X$.[8] We can manually check that the logical operators defined in Eq. (11) do commute with the stabilizer in Eq. (12). We will prove that this mapping preserves the fermionic algebra in Section. 3.

Given the stabilizer, we can provide the syndromes for all single-qubit Pauli errors, as shown in Fig. 2. We see that all single-qubit Pauli matrices have different syndromes, which means that we do not have any logical operators with weight 2. This implies a code distance of $d \geq 3$. Eq. (11) shows logical operators with weight 3, so we conclude that the code distance is $d = 3$. Based on the syndrome measurements, we can always correct any single-qubit error according to Fig. 2.

The four types of nearest-neighbor hopping terms, $\gamma_L\gamma'_R$, $\gamma_L\gamma_R$, $\gamma'_L\gamma'_R$, and $\gamma'_L\gamma_R$ have weights wt_i in the range $3 \leq \mathrm{wt}_i \leq 5$. Therefore, the modified bosonization has $(\mathrm{wt}_{\min}, \mathrm{wt}_{\max}) = (3,5)$ and a fermionic occupation term of weight 4. Compared to the original bosonization with $(\mathrm{wt}_{\min}, \mathrm{wt}_{\max}) = (2,6)$, the minimum weight is increased such that error correction can be performed, while the maximum weight of the hopping terms is decreased implying a reduction in the simulation cost.

Here we present the spinless Fermi-Hubbard Hamiltonian in 2d square lattice for $d = 3$ encoding as a concrete example. The 2d spinless Fermi-Hubbard Hamiltonian is

$$H = -t\sum_{\langle i,j\rangle}(c_i^\dagger c_j + \mathrm{h.c.}) + U\sum_{\langle i,j\rangle} c_i^\dagger c_i c_j^\dagger c_j\,, \tag{13}$$

where $\langle i,j\rangle$ represents the nearest-neighbor pair of vertices in the square lattice. We can write individual terms in Majorana basis such as $c_i^\dagger c_j + \mathrm{h.c.} = \frac{i}{2}(\gamma_i\gamma'_j - \gamma'_i\gamma_j)$, $c_i^\dagger c_i c_j^\dagger c_j = (\frac{1+i\gamma_i\gamma'_i}{2})(\frac{1+i\gamma_j\gamma'_j}{2})$.

[8] The stabilizer is derived from the identity $P_a P_c S_{58} S_{56} S_{25} S_{45} = 1$. After we map P_f and S_e to Pauli matrices using Eq. (11), it becomes $G_v^{d=3}$.

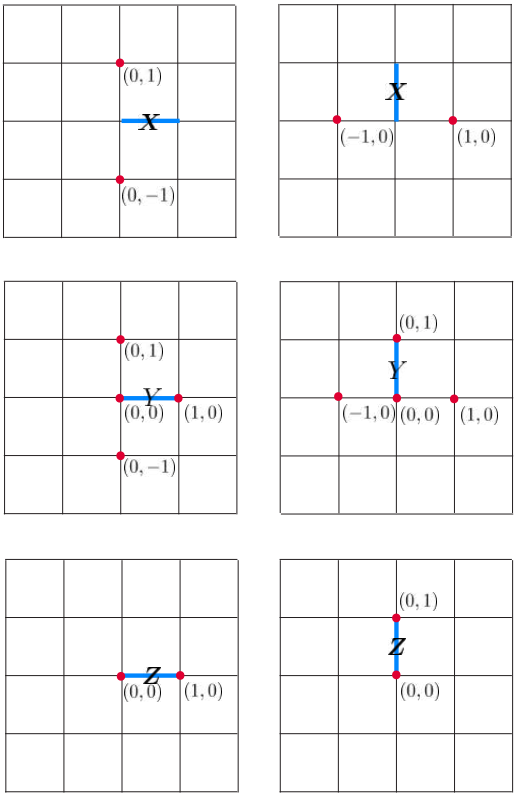
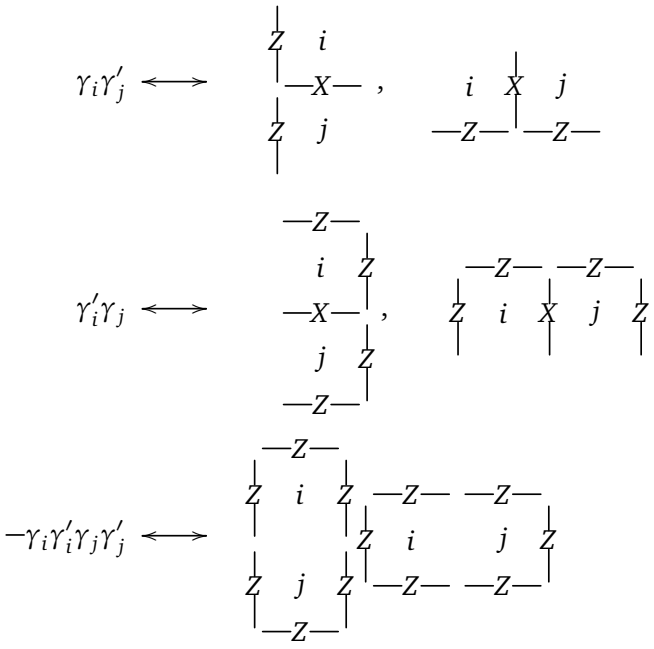


Figure 2: Syndromes of single-qubit errors for the bosonization with code distance $d = 3$. The red vertices v represent locations where the single-qubit error does not commute with the stabilizer G_v.

Next, we map the hopping terms and density-density interaction term to Pauli operators as follows



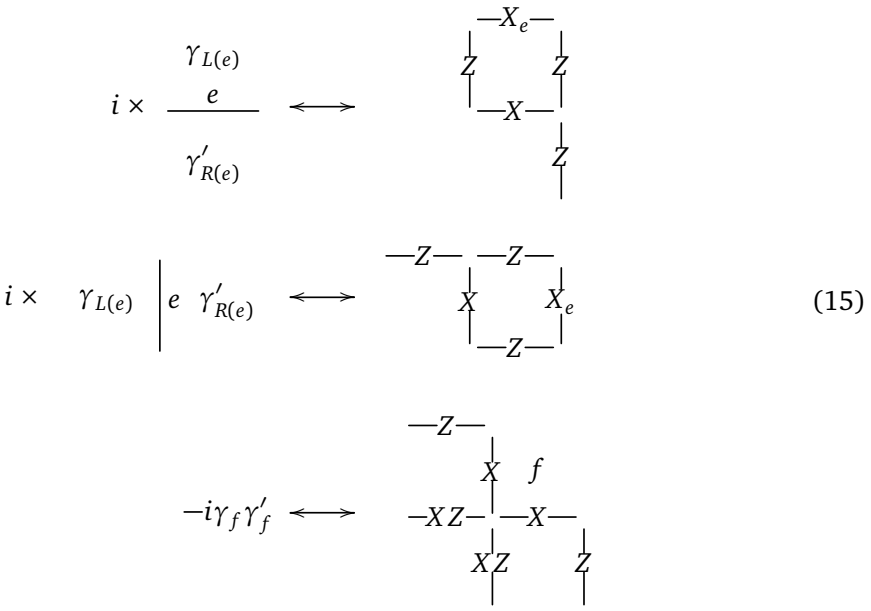
(14)

where two terms on the right-hand side correspond to vertical and horizontal (i, j). The hopping terms have weights ranging from 3 to 5, and the interaction term has a weight 6.

2.3 Bosonization with code distance $d = 4$

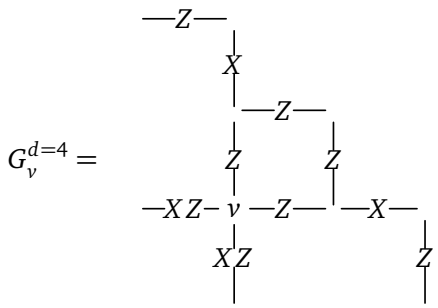
In this subsection, we provide a construction of an exact bosonization with code distance $d = 4$ as an intermediate step toward $d = 5$. Since its code distance is $d = 4$, which is even, this code (like the $d = 3$ code) can only correct the $\lfloor \frac{d-1}{2} \rfloor = 1$ Pauli error. However, for error defection, Pauli errors up to weight 3 can be observed from the stabilizer syndrome measurements.

The mapping can be described as



(15)

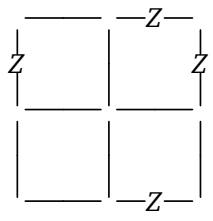
The stabilizer becomes



(16)

We can check that the logical operators in Eq. (15) commute with the stabilizer in Eq. (16). The proof of the equivalence between the even fermionic algebra and this stabilizer code will be shown in Section. 3.

The syndromes for all single-qubit Pauli errors are provided in Fig. 3. From the generators of the logical operators in Eq. (15), we may be tempted to conclude that the code distance is $d = 5$ because the minimum weight is 5. However, based on the syndromes in Fig. 3, we find that the following operator is logical:



(17)

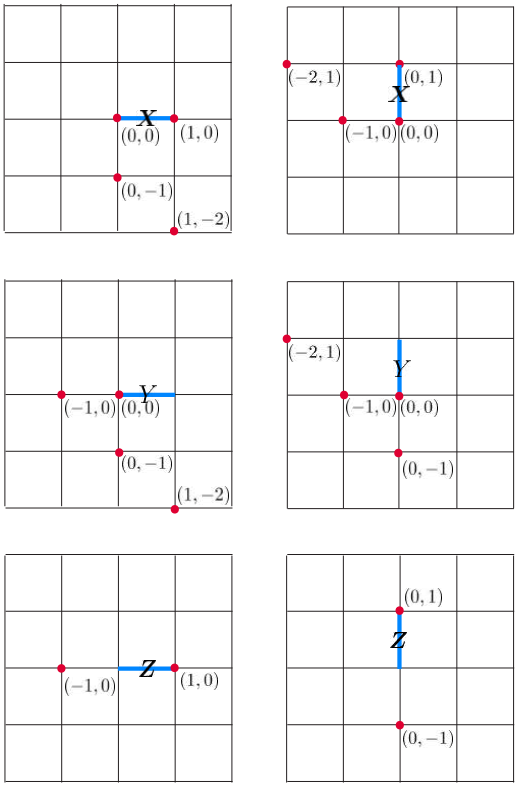


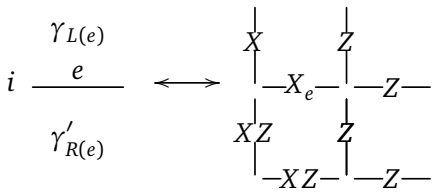
Figure 3: Syndromes of single-qubit errors for the bosonization with code distance $d = 4$. The red vertices v represent locations where the single-qubit error does not commute with the stabilizer $G_v^{d=4}$.

Since it does not commute with the terms in Eq. (15), this operator does not belong to the stabilizer group. Therefore, the code distance for this stabilizer code is $d \leq 4$. In Appendix A, we introduce the "syndrome matching" method to provide a lower bound for the code distance for a given stabilizer code. With this method, we check that the stabilizer code defined by Eq. (16) has a code distance of $d = 4$.

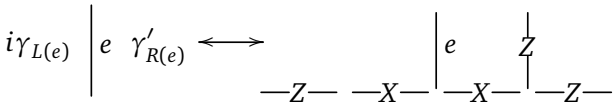
The minimum and maximum weights of the nearest-neighbor terms are ($\mathrm{wt}_{\mathrm{min}}$,$\mathrm{wt}_{\mathrm{max}}$)=(5,6) and the fermionic occupation term has weight 6, which means that all the operations are quite well-balanced. This code has an error-correcting property for any single-qubit error.

2.4 Bosonization with code distance $d = 5$

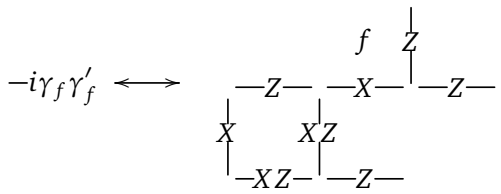
The bosonization map with code distance $d = 5$ is provided in this subsection. The generators of the even fermionic algebra are mapped to Pauli matrices as shown below:



(18)

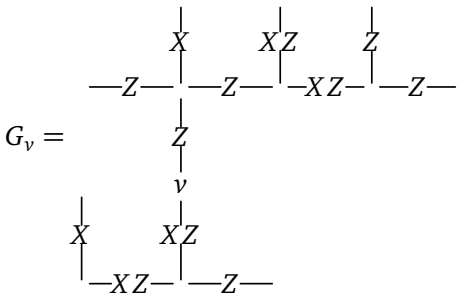


(19)



(20)

The stabilizer is



(21)

The minimum and maximum weights of the nearest-neighbor terms are $(\mathrm{wt}_{\min}, \mathrm{wt}_{\max}) = (5, 9)$ and the weight of the fermionic occupation term is 8. We use the "syndrome matching" method in Appendix A to confirm that $d = 5$. This code has an error-correcting property for any two-qubit error.

3 Stabilizer codes and the Pauli module

This section discusses the stabilizer code formalism and the Pauli module representation via Laurent polynomials. The Laurent polynomial method is reviewed in Section. 3.1. Then, in Section. 3.2, we discuss bosonization with distance $d = 3, 4, 5$ and the corresponding symplectic automorphisms. The searching algorithm for automorphisms is presented in Section. 3.3.

3.1 Review of the Laurent polynomial method for the Pauli algebra

We start by reviewing how any Pauli operator can be expressed as a vector over the **Laurent polynomial ring** $R = \mathbb{F}_2[x, y, x^{-1}, y^{-1}]$[9] as set out in Ref. [56]. First, we define X_{12}, Z_{12}, X_{14}, and Z_{14} in Fig. 1 as column vectors:

$$X_{12} = \begin{bmatrix} 1 \\ 0 \\ \hline 0 \\ 0 \end{bmatrix}, \quad Z_{12} = \begin{bmatrix} 0 \\ 0 \\ \hline 1 \\ 0 \end{bmatrix}, \quad X_{14} = \begin{bmatrix} 0 \\ 1 \\ \hline 0 \\ 0 \end{bmatrix}, \quad Z_{14} = \begin{bmatrix} 0 \\ 0 \\ \hline 0 \\ 1 \end{bmatrix}. \tag{22}$$

[9] This is the ring that consists of all linear combinations of polynomials involving x, x^{-1}, y, y^{-1} (and their powers) with coefficients in $\mathbb{F}_2$ (the field with two elements $\{0, 1\}$).

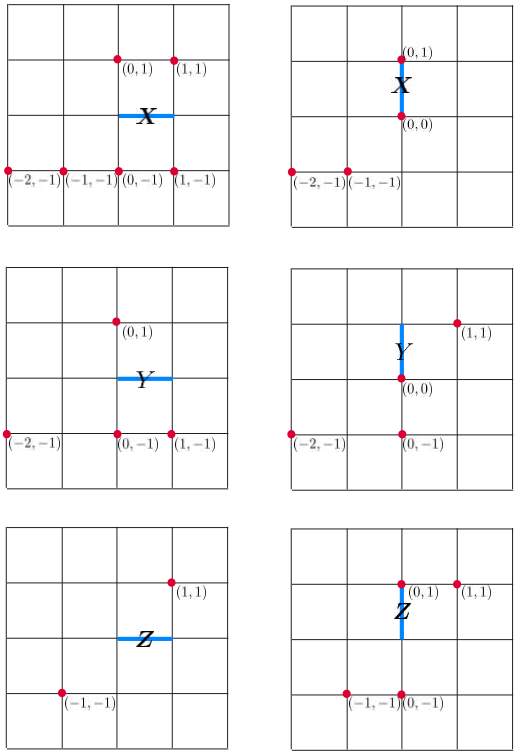


Figure 4: Syndromes of single-qubit errors for the $d = 5$ bosonization.

The Pauli Y can be written as a vector which is a sum of corresponding vectors of X and Z, such as

$$Y_{12} = \begin{bmatrix} 1 \\ 0 \\ 1 \\ 0 \end{bmatrix} . \tag{23}$$

We express the vector representation of the Pauli Y operator on an edge e as XZ along the same edge. It is important to note that in this representation, we quotient out the phase factor $\pm 1, \pm i$ associated with Pauli operators. For instance, we equate $Y, -Y, iY, -iY$ with Y. Our primary focus is on the commutation and anti-commutation properties of Pauli operators; thus, quotienting out the phase factor $\pm 1, \pm i$ does not impede our calculations. In practical applications, the phase factor can be efficiently tracked, as demonstrated in the Gottesman-Knill theorem [61, 62].

All the other edges can be defined with the help of translation operators as follows. We use polynomials of x and y to represent translation in the x and y directions, respectively. For example,

$$Z_{78} = y^2 \begin{bmatrix} 0 \\ 0 \\ 1 \\ 0 \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \\ y^2 \\ 0 \end{bmatrix} , \quad X_{58} = xy \begin{bmatrix} 0 \\ 1 \\ 0 \\ 0 \end{bmatrix} = \begin{bmatrix} 0 \\ xy \\ 0 \\ 0 \end{bmatrix} . \tag{24}$$

More examples are included in Fig. 5.

Next, we introduce the **antipode map** that is an $\mathbb{F}_2$-linear map from R to R defined by

$$x^a y^b \to \overline{x^a y^b} := x^{-a} y^{-b} . \tag{25}$$

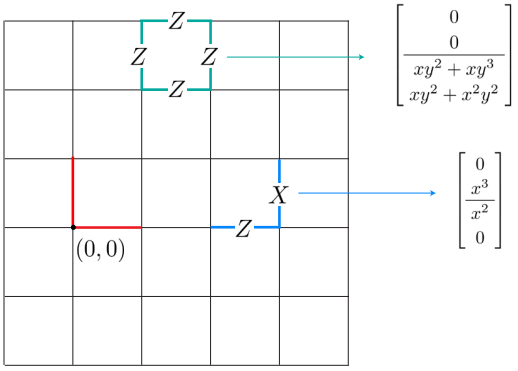


Figure 5: Examples of polynomial expressions for Pauli strings. The flux term (i.e., fermionic occupation) on a plaquette and the hopping term on an edge are both shown. The factors such as x^2y^2 and x^2 represent the locations of the operators relative to the origin.

To determine whether two Pauli operators represented by vectors v_1 and v_2 commute or anti-commute, we define the dot product as

$$v_1 \cdot v_2 = \overline{v}_1^T \Lambda v_2 \,, \tag{26}$$

where T is the transpose operation on a matrix and

$$\Lambda = \left[\begin{array}{cc|cc} 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \\ \hline -1 & 0 & 0 & 0 \\ 0 & -1 & 0 & 0 \end{array}\right] , \tag{27}$$

is the matrix representation of the standard **symplectic bilinear form**. Notice that -1 is the same as 1 because we are working over the $\mathbb{Z}_2$ field. For simplicity, we denote $\overline{(\cdots)}^T$ as $(\cdots)^\dagger$.

The two operators v_1 and v_2 commute if and only if the constant term of $v_1 \cdot v_2$ is zero. For example, we calculate the dot products

$$X_{12} \cdot Z_{12} = 1 \,, \qquad X_{58} \cdot Z_{14} = x^{-1}y^{-1} \,, \tag{28}$$

and, therefore, X_{12} and Z_{12} anti-commute, whereas X_{58} and Z_{14} commute (their dot product only has a non-constant term $x^{-1}y^{-1}$). Furthermore, the physical meaning of $X_{58} \cdot Z_{14} = x^{-1}y^{-1}$ is that the shifting of X_{58} in $-x$ and $-y$ directions by 1 step will anti-commute with Z_{14}. A translationally invariant stabilizer code forms an R-submodule[10] V such that

$$v_1 \cdot v_2 = v_1^\dagger \Lambda v_2 = 0 \,, \quad \forall v_1, v_2 \in V \,. \tag{29}$$

We now study the automorphisms A of the symplectic form Λ:

$$(Av_1) \cdot (Av_2) = v_1 \cdot v_2 \,, \quad \forall v_1, v_2 \in V \,. \tag{30}$$

[10] The R-submodule is similar to a subspace of a vector space, but the entries of the vector are in the ring $R = \mathbb{F}_2[x, y, x^{-1}, y^{-1}]$. In a ring, the inverse element may not exist. This is the distinction between a module and a vector space.

This is equivalent to $A^\dagger \Lambda A = \Lambda$. All matrices A satisfying this equation form the **symplectic group**. We divide A into 2×2 blocks $A = \left[\begin{array}{c|c} a & b \\ \hline c & d \end{array}\right]$, the automorphism condition becomes

$$\left[\begin{array}{c|c} a^\dagger & c^\dagger \\ \hline b^\dagger & d^\dagger \end{array}\right]\left[\begin{array}{c|c} 0 & I \\ \hline -I & 0 \end{array}\right]\left[\begin{array}{c|c} a & b \\ \hline c & d \end{array}\right] = \left[\begin{array}{c|c} 0 & I \\ \hline -I & 0 \end{array}\right] \tag{31}$$

$$\Rightarrow \quad a^\dagger d - c^\dagger b = I\,, \quad a^\dagger c = c^\dagger a\,, \quad b^\dagger d = d^\dagger b\,.$$

Examples of the automorphism A are

$$\begin{aligned} S &= \left[\begin{array}{c|c} I & 0 \\ \hline c & I \end{array}\right], \quad \text{where } c \in \mathrm{Mat}_2[R], \text{ and } c^\dagger = c\,, \\ H &= \left[\begin{array}{c|c} 0 & I \\ \hline -I & 0 \end{array}\right], \\ C &= \left[\begin{array}{cc|cc} 1 & 0 & 0 & 0 \\ r & 1 & 0 & 0 \\ \hline 0 & 0 & 1 & \bar{r} \\ 0 & 0 & 0 & 1 \end{array}\right], \quad \text{where } r \in R\,. \end{aligned} \tag{32}$$

$\mathrm{Mat}_2[R]$ consists of all 2×2 matrices with entries in $R = \mathbb{F}_2[x, y, x^{-1}, y^{-1}]$. The generators of the symplectic group are discussed in Appendix B. We have selected sixteen elementary automorphisms, denoted as $A_1, A_2, \cdots, A_{16}$, from the symplectic group. These automorphisms can be expressed by the conjugation of a unitary operator, i.e., the effect of the automorphism is

$$P \xrightarrow{A} UPU^\dagger\,. \tag{33}$$

The corresponding unitary circuits for the sixteen elementary automorphisms are illustrated in Fig. 6.

3.2 New stabilizer codes developed from automorphisms

First, we reformulate the original bosonization introduced in Section 2.1 and incorporate it into the Pauil module. For simplicity, we will write x^{-1} and y^{-1} as $\overline{x}$ and $\overline{y}$, respectively. The original hopping operators U_e in Eq. (5) can be written as

$$U_1 = \left[\begin{array}{c} 1 \\ 0 \\ \hline 0 \\ \overline{y} \end{array}\right], \quad U_2 = \left[\begin{array}{c} 0 \\ 1 \\ \hline \overline{x} \\ 0 \end{array}\right], \tag{34}$$

where U_1 represents U_e on the horizontal edge and U_2 represents U_e on the vertical edge. The flux term W_f in (6) is written as

$$W = \left[\begin{array}{c} 0 \\ 0 \\ \hline 1+y \\ 1+x \end{array}\right]. \tag{35}$$

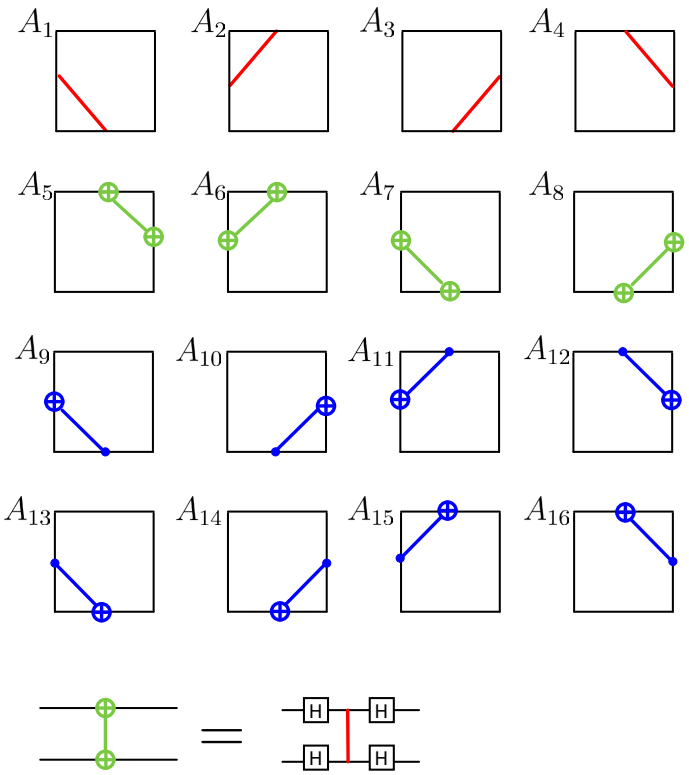


Figure 6: The circuit representations of sixteen elementary automorphisms. Red lines represent the CZ gates, green lines represent the $H^{\otimes 2}(CZ)H^{\otimes 2}$ gates, and blue lines represent the $CNOT$ gates.

The stabilizer G_v in Eq. (9) corresponds to the vector

$$G = \left[\begin{array}{c} 1+\overline{x} \\ 1+\overline{y} \\ \hline 1+y \\ 1+x \end{array}\right]. \tag{36}$$

Now, we will apply automorphisms on these vectors to generate new stabilizer codes.

3.2.1 Automorphism for code distance $d = 3$

We consider the simplest automorphism

$$A_1 = \left[\begin{array}{cc|cc} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ \hline 0 & 1 & 1 & 0 \\ 1 & 0 & 0 & 1 \end{array}\right], \tag{37}$$

which modifies the Pauli operator X_e as

$$A_1 \left[\begin{array}{c} 1 \\ 0 \\ \hline 0 \\ 0 \end{array}\right] = \left[\begin{array}{c} 1 \\ 0 \\ \hline 0 \\ 1 \end{array}\right], \quad A_1 \left[\begin{array}{c} 0 \\ 1 \\ \hline 0 \\ 0 \end{array}\right] = \left[\begin{array}{c} 0 \\ 1 \\ \hline 1 \\ 0 \end{array}\right]. \tag{38}$$

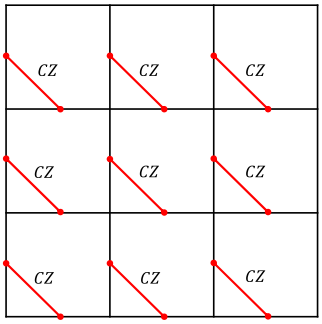


Figure 7: Clifford circuit corresponding to automorphism A_1.

Pictorially, this is equivalent to

$$X_e = \left\{ \begin{array}{ll} -X_e- & \xrightarrow{A_1} \begin{array}{c} Z \\ | \\ -X_e- \end{array} \\ \begin{array}{c} X_e \\ | \end{array} & \xrightarrow{A_1} \begin{array}{c} X_e \\ | \\ -Z- \end{array} \end{array} \right. \tag{39}$$

Notice that Z_e is unchanged under this automorphism. This automorphism corrsponds to the Clifford circuit shown in Fig. 7.

Now we apply A_1 on the logical operators U_1, U_2, and W and the stabilizer G:

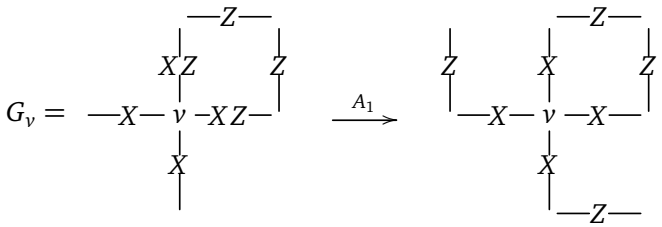
$$A_1 U_1 = \begin{bmatrix} 1 \\ 0 \\ \hline 0 \\ 1+\overline{y} \end{bmatrix}, \qquad A_1 U_2 = \begin{bmatrix} 0 \\ 1 \\ \hline 1+\overline{x} \\ 0 \end{bmatrix}, \tag{40}$$

$$A_1 W = \begin{bmatrix} 0 \\ 0 \\ \hline 1+y \\ 1+x \end{bmatrix}, \qquad A_1 G = \begin{bmatrix} 1+\overline{x} \\ 1+\overline{y} \\ \hline y+\overline{y} \\ x+\overline{x} \end{bmatrix}. \tag{41}$$

The automorphism A_1 applied on U_e can be visualized as

$$U_e = \left\{ \begin{array}{ll} \begin{array}{c} X_e \\ | \\ -Z- \end{array} & \xrightarrow{A_1} \begin{array}{c} X_e \\ | \\ -Z-|-Z- \end{array} \\ -X_e- & \\ \begin{array}{c} | \\ Z \end{array} & \xrightarrow{A_1} \begin{array}{c} Z \\ | \\ -X_e- \\ | \\ Z \end{array} \end{array} \right. \tag{42}$$

The flux term Eq. (6) is unchanged. The automorphism A_1 applied on the stabilizer G_v is

$G_v =$  (43)

This is the bosonization with code distance $d = 3$ introduced in Section 2.2. Since we applied the automorphism of the Pauli module on the original bosonization, the logical operators satisfy the same algebra. Therefore, we conclude that this new stabilizer code is a valid way to simulate fermions.

3.2.2 Automorphism for code distance $d = 4$

In this section, we consider a slightly more complicated automorphism A':[11]

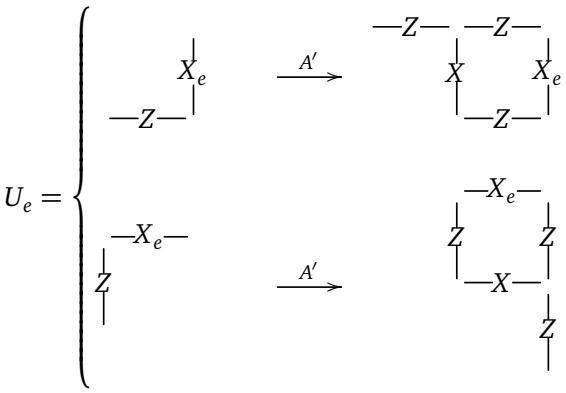
$$A' = \left[\begin{array}{cc|cc} 1 & 0 & 0 & 1 \\ 0 & 1 & 1 & 0 \\ \hline 0 & \overline{x}y & 1+\overline{x}y & 0 \\ x\overline{y} & 0 & 0 & 1+x\overline{y} \end{array}\right]. \tag{44}$$

One can easily check that A' indeed satisfies the condition in Eq. (31) for being an automorphism. Applying A' on the logical operators U_1, U_2, and W and the stabilizer G, we get

$$A'U_1 = \left[\begin{array}{c} 1+\overline{y} \\ 0 \\ \hline 0 \\ x\overline{y}^2 + x\overline{y} + \overline{y} \end{array}\right], \qquad A'U_2 = \left[\begin{array}{c} 0 \\ 1+\overline{x} \\ \hline \overline{x} + \overline{x}y + \overline{x}^2 y \\ 0 \end{array}\right], \tag{45}$$

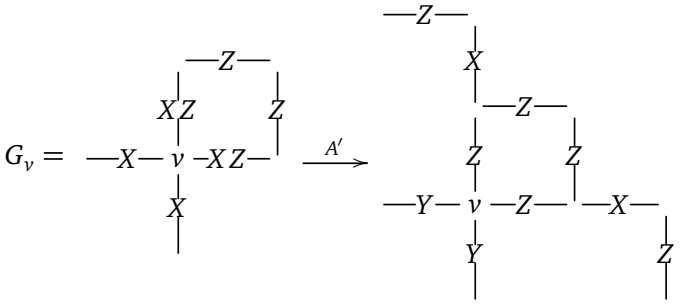
$$A'W = \left[\begin{array}{c} 1+x \\ 1+y \\ \hline 1+y+\overline{x}+\overline{x}y^2 \\ 1+x+\overline{y}+\overline{y}x^2 \end{array}\right], \qquad A'G = \left[\begin{array}{c} x+\overline{x} \\ y+\overline{y} \\ \hline 1+y+\overline{x}+\overline{x}y^2 \\ 1+x+\overline{y}+\overline{y}x^2 \end{array}\right]. \tag{46}$$

The operators $A'(U_e)$ can be depicted as

$U_e =$  (47)

[11] $A' = A_4 A_7$ in terms of the elementary automorphisms defined in Appendix B.

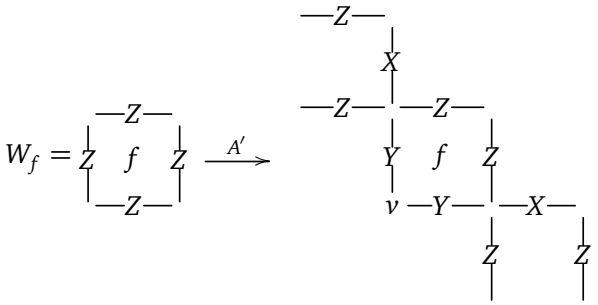
The stabilizer $A'(G_v)$ is



(48)

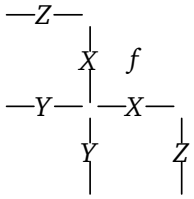
The logical operator U_e and the stabilizer G_v are mapped on the bosonization with code distance $d = 4$ introduced in Section 2.3.

Finally, the flux term under the automorphism A' becomes



(49)

This term can be simplified by multiplying it by $A'(G_v)$, which does not change the effective logical operation. The resulting flux term is



(50)

which matches Eq. (15) in Section 2.3.

3.2.3 Automorphism for code distance $d = 5$

We introduce another automorphism A'':[12]

$$A'' = \left[\begin{array}{cc|cc} 1 & \overline{x} & x & 1 \\ 1 & \overline{x}+1 & 1+x & 1 \\ \hline x & \overline{x}+1 & \overline{x}+1+x+x^2 & 1+x \\ x & 1 & x+x^2 & 1+x \end{array}\right]. \tag{51}$$

[12] $A'' = A_9 A_3 A_7 A_{14}$ in terms of the elementary automorphisms defined in Appendix B.

Again it is easy to check that it satisfies the condition in Eq. (31) for being an automorphism. Applying A'' on the logical operators U_1, U_2, and W and the stabilizer G, we get

$$A''U_1 = \begin{bmatrix} 1+\overline{y} \\ 1+\overline{y} \\ \overline{y}+x\overline{y}+x \\ \overline{y}+x\overline{y}+x \end{bmatrix}, \qquad A''U_2 = \begin{bmatrix} 1+\overline{x} \\ 0 \\ \overline{x^2}+x \\ x \end{bmatrix}, \tag{52}$$

$$A''(W+G) = \begin{bmatrix} \overline{xy}+1 \\ \overline{xy}+\overline{y} \\ \overline{xy}+\overline{y}+\overline{x}+x \\ \overline{y}+x \end{bmatrix}, \qquad A''G = \begin{bmatrix} \overline{xy}+xy \\ \overline{xy}+\overline{y}+y+xy \\ \overline{xy}+\overline{y}+\overline{x}y+y+xy+x^2y \\ \overline{y}+1+xy+x^2y \end{bmatrix}. \tag{53}$$

The $A''(U_e)$ operators can be depicted as Eq. (18) and (19). Here, we choose $A''(W+G)$ as our flux operator shown in Eq. (20) because it has a lower weight $\mathrm{wt}[A''(W+G)] < \mathrm{wt}(A''W)$. The pictorial representation of stabilizer $A''G$ is Eq. (21).

3.3 Searching algorithm for automorphisms

In this subsection, we describe how we find automorphisms with code distances $d = 3, 4, 5, 6,$ and 7. The automorphisms A_1 in Eq. (37), A' in Eq. (44), and A'' in Eq. (51) correspond to the examples for $d = 3$, $d = 4$, and $d = 5$, respectively. We will show other examples with different code distances.

First, we consider sixteen elementary automorphisms $A_1, A_2, \cdots, A_{16}$ (shown in Appendix B), which attach no more than one new Pauli matrix to the original Pauli matrix. For example, the A_1 automorphism attaches one Z to X_e, as shown in Eq. (38). Given the sixteen elementary automorphisms, their product $A_{i_1}A_{i_2}A_{i_3}A_{i_4}\cdots$ with $i_n \in \{1, 2, \cdots, 16\}$ is also an automorphism. (We note that these elementary automorphisms do not generate all automorphisms.) We find that the product of five elementary automorphisms $A_{i_1}A_{i_2}A_{i_3}A_{i_4}A_{i_5}$ is sufficient to generate the code distance $d = 7$; therefore, we focus on products with five or fewer elementary automorphisms. We now describe how we search for automorphisms with large code distances:

1. We write down the bosonization of all the nearest-neighbor and on-site terms generated by S_e and P_f. They include the automorphism acting on U_1, U_2, W, U_1+W, $U_1+\overline{y}W$, $U_1+\overline{y}W+W$, U_2+W, $U_2+\overline{x}W$, $U_2+\overline{x}W+W$.[13]

2. We use the minimum weight of bosonization of nearest-neighbor hopping terms to roughly estimate the code distance of a given bosonization (automorphism) A.

3. We choose some candidate automorphisms with an appropriate minimum weight d, to find a bosonization with desired code distance d.

4. We apply the syndrome matching method shown in Appendix A to the candidate automorphisms. By applying syndrome matching, we find a lower bound of their code distances.

5. We apply the syndrome matching method for distance $d+1$ to an automorphism $\widetilde{A}$ with a lower bound d. If syndrome matching for distance $d+1$ returns a logical operator with no syndrome, we conclude that $\widetilde{A}$ is a bosonization with code distance d.

Applying the syndrome-matching method, we found automorphisms to generate exact bosonization with $d = 3, 4, 5, 6, 7$ (see Table 2).

[13] The stabilizer G can be added to any term.

Table 2: The possible automorphisms for different code distances. The numbers inside parentheses are the minimum and maximum weights of the logical operators for the nearest-neighbor terms. For example, $A_9A_3A_7A_{14}$ has the minimum logical weight 5 and the maximum logical weight 9.

Code distance	Automorphisms
3	A_1 (3,5)
4	A_4A_7 (5,6), $A_2A_7A_1$ (4,6)
5	$A_9A_3A_7A_{14}$ (5,9)
6	$A_1A_5A_{14}A_1(6,13)$, $A_4A_9A_{16}A_{11}(7,17)$
7	$A_1A_{11}A_5A_{14}A_9(7,23)$

4 Discussion

This work introduces a method that employs Laurent polynomials and symplectic automorphisms as efficient classical computational tools in the search for fermion-to-qubit mappings with error correction capabilities. One significant advantage of this method lies in its ability to generate equivalent mappings with higher code distances while preserving the code rates. This is possible due to the established equivalence between various 2d fermion-to-qubit mappings, as shown in Ref. [48].

In recent years, it has been shown that Laurent polynomials do not only serve as an analytical tool to design and characterize quantum code [56, 57, 60, 63–65], numerical algorithm inspired by Laurent polynomials are also proposed, for example, searching 3d fracton phases [66]. In this work, we demonstrate the effectiveness of the Laurent polynomial in searching quantum error-correcting codes. The Laurent polynomials and symplectic automorphisms serve a dual purpose in our method: They are instrumental in the analytical derivation of quantum codes and valuable for the numerical studies of these codes. To illustrate the effectiveness of our approach, we demonstrate this method through examples of codes with distances $d = 3, 4, 5$ and further extend our constructions up to $d = 7$. Additionally, we present general algorithms designed to systematically search for and verify fermion-to-qubit mappings with elevated code distances. It is noteworthy that our proposed method is not exclusive to fermion-to-qubit mappings; it is also applicable to regular qubit stabilizer codes, which are utilized to encode logical qubits.

In the context of implementing higher-distance error-correcting codes, which are realized through exact bosonization (with $d = 2$) by Clifford deformations, we can prepare the codewords for these codes. This is achieved by applying the appropriate Clifford circuit to the codewords of the exact bosonization ($d = 2$). The process involves the following steps:

1. Generating a product state in the fermionic Fock basis from the toric code ground state by exciting fermions $\epsilon = e \times m$. This state is a codeword of the exact bosonization ($d = 2$).

2. Transforming this codeword using the Clifford unitary corresponding to the automorphism A.

3. The result is the codeword for the higher-distance exact bosonization, preserving the same logical information. In particular, the Clifford circuit used is geometrically local and translationally invariant, ensuring that a shallow, $O(1)$-depth, geometrically local Clifford circuit does not spread errors.

This method provides a framework for developing higher-distance error-correcting codes essential for fermionic quantum simulation, simultaneously underscoring the necessity of efficient decoders for effective error correction. Given that exact bosonization originates from

the toric code, we anticipate that the minimum weight perfect matching approach still works. Apart from quantum error correction, which includes code construction and decoder development, optimizing efficient and fault-tolerant logical operations remains a critical challenge. Exploring efficient decoders and fault-tolerant operations is left for future research.

Acknowledgment

Y.-A.C wants to thank Zhang Jiang for the discussions that inspired the main idea of this paper. Y.-A.C is also grateful to Nat Tantivasadakarn for teaching the polynomial method and its application to the 2d bosonization. We also offer our thanks to Victor Albert, Riley Chien, Daniel Gottesman, Michael Gullans, Mohammad Hafezi, and Ben Reichardt for engaging in very useful discussions with us.

Note added.—As this work was being completed, we also became aware of an independent work, Ref. [67], using a similar polynomial technique to study fermionic quantum simulation from a different perspective. The other recent work Ref. [68] studied an error-mitigation scheme in fermionic encodings. The recent development of programmable neutral atom arrays in Ref. [69] can simulate fermionic modes directly, which avoids using any fermion-to-qubit mapping.

Funding information Y.-A.C. is supported by the JQI fellowship. A.V.G. was supported in part by NSF QLCI (award No. OMA-2120757), DoE ASCR Accelerated Research in Quantum Computing program (award No. DE-SC0020312), DoE QSA, the DoE ASCR Quantum Testbed Pathfinder program (award No. DE-SC0019040), NSF PFCQC program, AFOSR, ARO MURI, AFOSR MURI, and DARPA SAVaNT ADVENT. Y.X. is supported by ARO W911NF-15-1-0397, National Science Foundation QLCI grant OMA-2120757, AFOSR-MURI FA9550-19-1-0399, Department of Energy QSA program. This work is supported by the Laboratory for Physical Sciences through the Condensed Matter Theory Center.

A Syndrome-matching method for finding code distances

This appendix presents an algorithm to find the code distance for a given stabilizer code. We call it the "syndrome matching" method. Specifically, given an integer n, we describe an algorithm to determine whether the code distance d satisfies $d > n$. The algorithm is as follows.

1. We first choose one vertex to be the origin $(0,0)$. We then apply a single Pauli from X_1, Y_1, Z_1, X_2, Y_2, Z_2 (acting on qubits located on edges $(0,0)\to(1,0)$ and $(0,0)\to(0,1)$). We have 6 cases, each of which has its own syndrome vertices, i.e., a set of vertices v that violate the stabilizer G_v. For a single-qubit error, the syndrome set is an ordered set $V^{(1)} = \{(x_1^{(1)}, y_1^{(1)}), (x_2^{(1)}, y_2^{(1)}), ...\}$, ordered by $x_i^{(1)}$: $x_1^{(1)} \le x_2^{(1)} \le \ldots$. If two vertices $i, i+1$ have the same $x_i^{(1)} = x_{i+1}^{(1)}$, they are ordered by $y_i^{(1)} < y_{i+1}^{(1)}$.

2. Ensure that the operator is logical. For this to be the case, all syndrome vertices should vanish. Therefore, we select the first syndrome vertex $(x_1^{(k)}, y_1^{(k)}) \in V^{(k)}$. Then we enumerate all choices of a Pauli matrix on an edge different from the Pauli matrix selected in the previous step(s), such that it cancels the syndrome at $(x_1^{(k)}, y_1^{(k)})$. This operation may generate other syndrome vertices. The syndrome vertices $V^{(k)}$ are updated due to this new Pauli matrix. At this stage, the operator has one more Pauli matrix, and a new

ordered set for the syndrome vertices $V^{(k+1)}$. If the syndrome set $V^{(k+1)}$ is empty, this operator is logical. If the operator does not belong to the stabilizer group,[14] this means that we have found the nontrivial logical operator with the minimum weight. This minimum weight is the code distance and, therefore, we stop the algorithm. Otherwise, we continue.

3. Repeat steps 1 and 2 until the algorithm stops automatically or all cases with operators containing n Pauli matrices have been considered, i.e., all $V^{(n)}$ are checked. If the algorithm stops automatically, it will return the value of the code distance. If the algorithm stops by considering all the cases with n Pauli matrices, we conclude that code distance satisfies $d > n$.

B Elementary automorphisms

This section demonstrates the transformation rules of Pauli matrices for sixteen elementary automorphisms used in our numerical search algorithm. First, according to Ref. [65], the symplectic group of vectors over the Laurent polynomial ring $R = \mathbb{F}_2[x, y, x^{-1}, y^{-1}]$ is generated by the following 4×4 matrices, which form the **elementary symplectic group**. Below $a \in R^\times$ is any monomial in R.

$$\begin{aligned} \text{Hadamard:} &\quad E_{i,i+2}(-1)E_{i+2,i}(1)E_{i,i+2}(-1), &&\text{where } 1 \le i \le 2, \\ \text{control-Phase:} &\quad E_{i+2,j}(a)E_{j+2,i}(\bar{a}), &&\text{where } 1 \le i, j \le 2, \\ \text{control-Not:} &\quad E_{i,j}(a)E_{j+2,i+2}(-\bar{a}), &&\text{where } 1 \le i \ne j \le 2. \end{aligned} \tag{B.1}$$

where the matrix $E_{i,j}(a)$ for any $a \in R$ is defined as

$$\big[E_{i,j}(a)\big]_{\mu\nu} = \delta_{\mu\nu} + \delta_{\mu i}\delta_{\nu j}a\,, \quad \text{where } \delta \text{ is the Kronecker delta.} \tag{B.2}$$

More explicitly, the Hamamard gates for $i = 1$ and $i = 2$ are

$$\left[\begin{array}{cc|cc} 0 & 0 & -1 & 0 \\ 0 & 1 & 0 & 0 \\ \hline 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 \end{array}\right], \quad \left[\begin{array}{cc|cc} 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & -1 \\ \hline 0 & 0 & 1 & 0 \\ 0 & 1 & 0 & 0 \end{array}\right]. \tag{B.3}$$

The control-Phase gates for $(i, j) = (1,1), (1,2), (2,1), (2,2)$ are

$$\left[\begin{array}{cc|cc} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ \hline a+\bar{a} & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{array}\right], \left[\begin{array}{cc|cc} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ \hline 0 & a & 1 & 0 \\ \bar{a} & 0 & 0 & 1 \end{array}\right], \left[\begin{array}{cc|cc} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ \hline 0 & \bar{a} & 1 & 0 \\ a & 0 & 0 & 1 \end{array}\right], \left[\begin{array}{cc|cc} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ \hline 0 & 0 & 1 & 0 \\ 0 & a+\bar{a} & 0 & 1 \end{array}\right]. \tag{B.4}$$

The control-Not gates for $(i, j) = (1,2), (2,1)$ are

$$\left[\begin{array}{cc|cc} 1 & a & 0 & 0 \\ 0 & 1 & 0 & 0 \\ \hline 0 & 0 & 1 & 0 \\ 0 & 0 & -\bar{a} & 1 \end{array}\right], \quad \left[\begin{array}{cc|cc} 1 & 0 & 0 & 0 \\ a & 1 & 0 & 0 \\ \hline 0 & 0 & 1 & -\bar{a} \\ 0 & 0 & 0 & 1 \end{array}\right]. \tag{B.5}$$

[14] A way to check if a Pauli string operator O belongs to the stabilizer group is by computing $O \cdot (AW)$ and $O \cdot (AU_{1,2})$, where AW and $AU_{1,2}$ generate the full logical space since W and $U_{1,2}$ are the original generators in the exact bosonization and we apply an automorphism A on them. $O \cdot (AW) = O \cdot (AU_{1,2}) = 0$ if and only if the operator O commutes with all logical operators, which means $O \in \mathcal{G}$ ($\mathcal{G}$ is the stabilizer group).

From the elementary symplectic group above, we choose the following automorphisms $A_1,...,A_{16}$, corresponding to applying nearest-neighbor two-qubit Clifford gates.

$$A_1 = \left[\begin{array}{cc|cc} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ \hline 0 & 1 & 1 & 0 \\ 1 & 0 & 0 & 1 \end{array}\right], \qquad A_2 = \left[\begin{array}{cc|cc} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ \hline 0 & y & 1 & 0 \\ \bar{y} & 0 & 0 & 1 \end{array}\right], \tag{B.6}$$

$$A_3 = \left[\begin{array}{cc|cc} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ \hline 0 & \overline{x} & 1 & 0 \\ x & 0 & 0 & 1 \end{array}\right], \qquad A_4 = \left[\begin{array}{cc|cc} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ \hline 0 & \overline{x}y & 1 & 0 \\ x\overline{y} & 0 & 0 & 1 \end{array}\right], \tag{B.7}$$

$$A_5 = \left[\begin{array}{cc|cc} 1 & 0 & 0 & \overline{x}y \\ 0 & 1 & x\overline{y} & 0 \\ \hline 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{array}\right], \qquad A_6 = \left[\begin{array}{cc|cc} 1 & 0 & 0 & y \\ 0 & 1 & \overline{y} & 0 \\ \hline 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{array}\right], \tag{B.8}$$

$$A_7 = \left[\begin{array}{cc|cc} 1 & 0 & 0 & 1 \\ 0 & 1 & 1 & 0 \\ \hline 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{array}\right], \qquad A_8 = \left[\begin{array}{cc|cc} 1 & 0 & 0 & \overline{x} \\ 0 & 1 & x & 0 \\ \hline 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{array}\right], \tag{B.9}$$

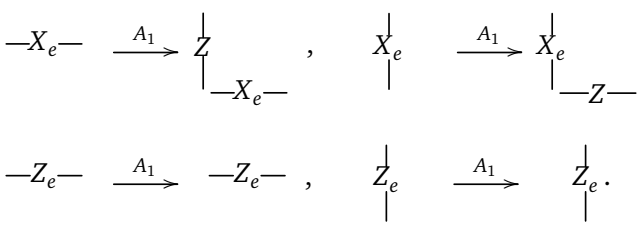
$$A_9 = \left[\begin{array}{cc|cc} 1 & 0 & 0 & 0 \\ 1 & 1 & 0 & 0 \\ \hline 0 & 0 & 1 & 1 \\ 0 & 0 & 0 & 1 \end{array}\right], \qquad A_{10} = \left[\begin{array}{cc|cc} 1 & 0 & 0 & 0 \\ x & 1 & 0 & 0 \\ \hline 0 & 0 & 1 & \overline{x} \\ 0 & 0 & 0 & 1 \end{array}\right], \tag{B.10}$$

$$A_{11} = \left[\begin{array}{cc|cc} 1 & 0 & 0 & 0 \\ \overline{y} & 1 & 0 & 0 \\ \hline 0 & 0 & 1 & y \\ 0 & 0 & 0 & 1 \end{array}\right], \qquad A_{12} = \left[\begin{array}{cc|cc} 1 & 0 & 0 & 0 \\ x\overline{y} & 1 & 0 & 0 \\ \hline 0 & 0 & 1 & \overline{x}y \\ 0 & 0 & 0 & 1 \end{array}\right], \tag{B.11}$$

$$A_{13} = \left[\begin{array}{cc|cc} 1 & 1 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ \hline 0 & 0 & 1 & 0 \\ 0 & 0 & 1 & 1 \end{array}\right], \qquad A_{14} = \left[\begin{array}{cc|cc} 1 & \overline{x} & 0 & 0 \\ 0 & 1 & 0 & 0 \\ \hline 0 & 0 & 1 & 0 \\ 0 & 0 & x & 1 \end{array}\right], \tag{B.12}$$

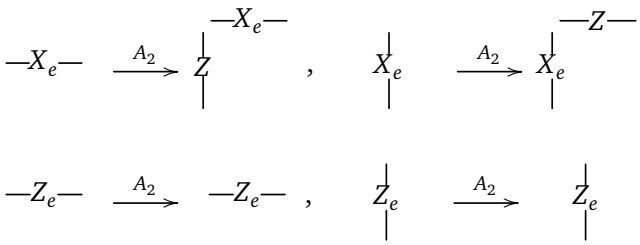
$$A_{15} = \left[\begin{array}{cc|cc} 1 & y & 0 & 0 \\ 0 & 1 & 0 & 0 \\ \hline 0 & 0 & 1 & 0 \\ 0 & 0 & \overline{y} & 1 \end{array}\right], \qquad A_{16} = \left[\begin{array}{cc|cc} 1 & \overline{x}y & 0 & 0 \\ 0 & 1 & 0 & 0 \\ \hline 0 & 0 & 1 & 0 \\ 0 & 0 & x\overline{y} & 1 \end{array}\right]. \tag{B.13}$$

The diagonal blocks of A_1 are identities, and its nontrivial part is the lower left block which attaches an extra Z to X_1 and X_2. By multiplying the polynomial vectors of X_1, X_2, Z_1, and Z_2 by automorphism A_1, we get the following terms:



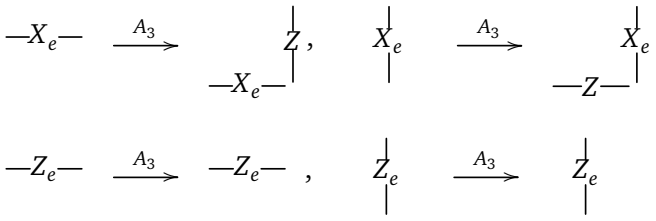
(B.14)

Similarly, we may multiply X_1, X_2, Z_1, and Z_2 by A_2, thereby obtaining



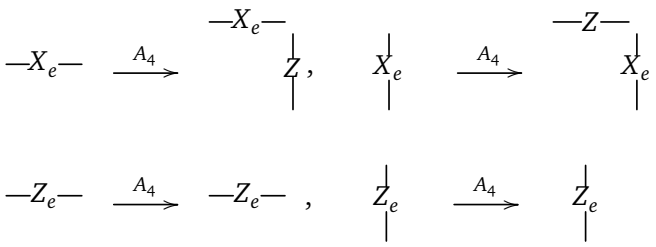
(B.15)

Following the same argument, we can obtain pictorial representations for the rest of the automorphisms: A_3:



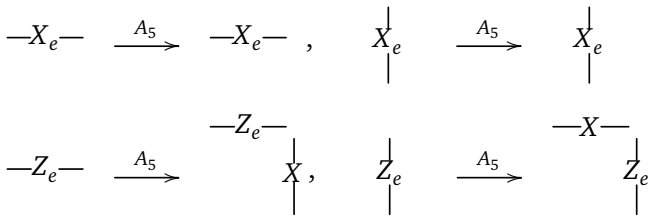
(B.16)

A_4:



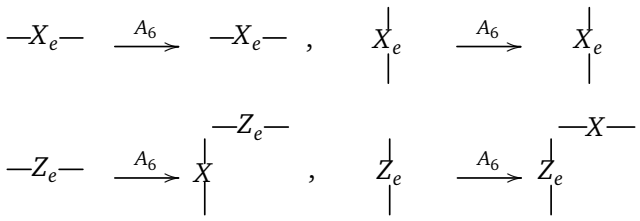
(B.17)

A_5:



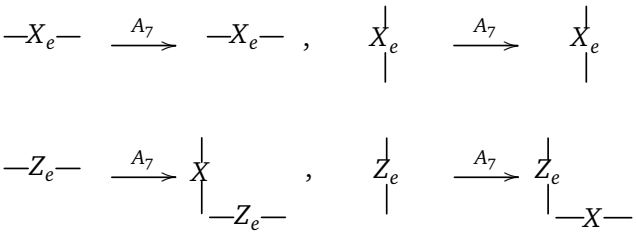
(B.18)

A_6:



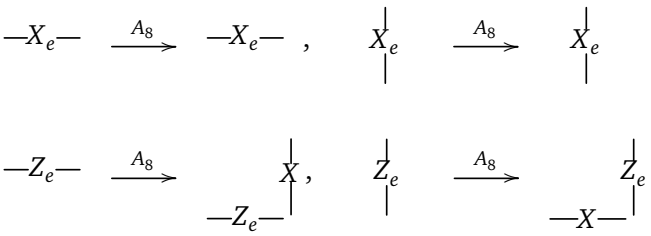
(B.19)

A_7:



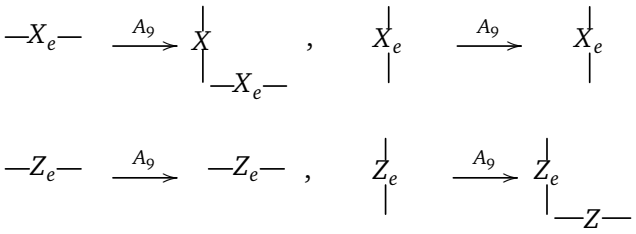
(B.20)

A_8:



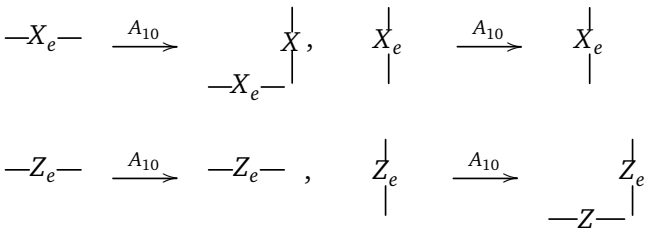
(B.21)

A_9:



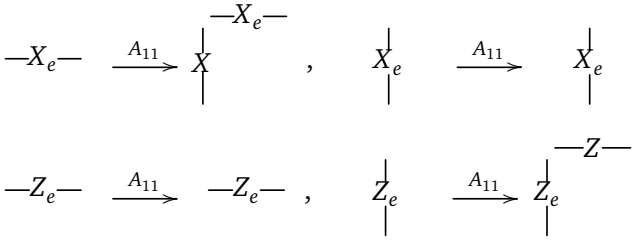
(B.22)

A_{10}:



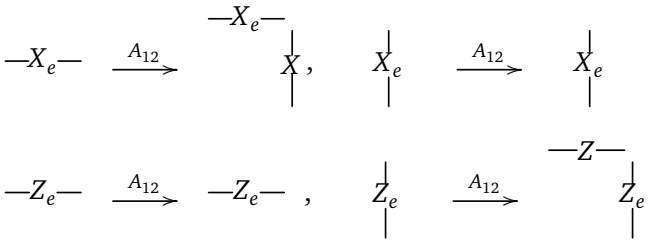
(B.23)

A_{11}:



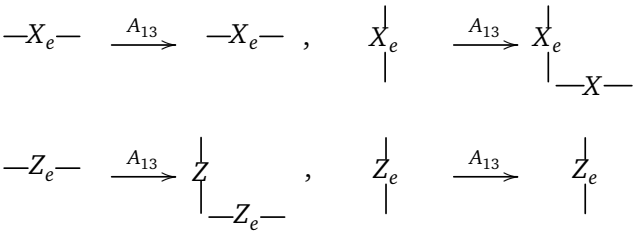
(B.24)

A_{12}:



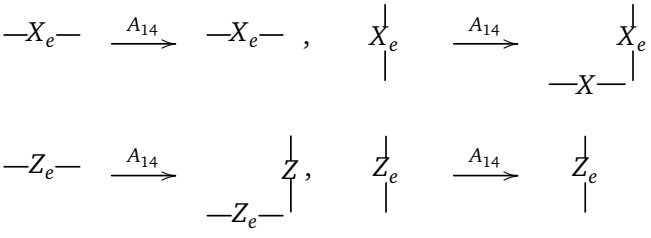
(B.25)

A_{13}:



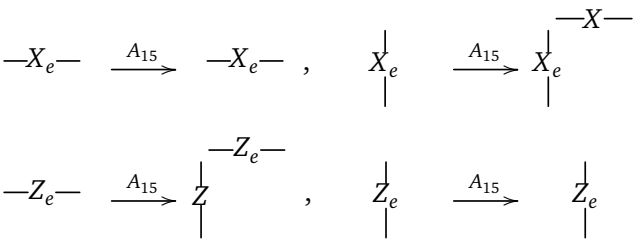
(B.26)

A_{14}:



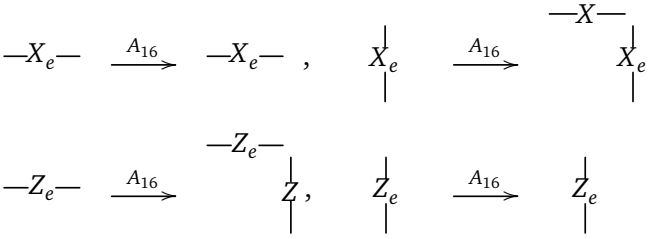
(B.27)

A_{15}:



(B.28)

A_{16}:



(B.29)

C Automorphisms for code distances $d = 6$ and $d = 7$

In this section, we show the explicit form of automorphisms $A^{d=6}$ and $A^{d=7}$ found by syndrome matching. The automorphism $A^{d=6} = A_1A_5A_{14}A_1$ has a code distance of 6:

$$A^{d=6} = \left[\begin{array}{cc|cc} 1+\overline{x}y & \overline{x}+y & y & \overline{x}y \\ 0 & 1+x\overline{y} & x\overline{y} & 0 \\ \hline 0 & x\overline{y} & 1+x\overline{y} & 0 \\ \overline{x}y & \overline{x}+x+y & x+y & 1+\overline{x}y \end{array}\right]. \tag{C.1}$$

By applying $A^{d=6}$ on logical operators U_1, U_2, W, and stabilizer G, we obtain their polynomial representations as follows:

$$A^{d=6}U_1 = \left[\begin{array}{c} \overline{x}+1+\overline{x}y \\ 0 \\ \hline 0 \\ \overline{y}+\overline{x}+\overline{x}y \end{array}\right], \tag{C.2}$$

$$A^{d=6}U_2 = \left[\begin{array}{c} \overline{x}+\overline{x}y+y \\ \overline{y}+x\overline{y}+1 \\ \hline \overline{y}+x\overline{y}+\overline{x} \\ \overline{x}+1+x+\overline{x}y+y \end{array}\right], \tag{C.3}$$

$$A^{d=6}W = \left[\begin{array}{c} \overline{x}y+y^2 \\ x\overline{y}+x \\ \hline x\overline{y}+1+x+y \\ 1+\overline{x}y+xy+y^2 \end{array}\right], \tag{C.4}$$

$$A^{d=6}G = \left[\begin{array}{c} \overline{xy}+\overline{x}^2y+y+y^2 \\ x\overline{y}^2+\overline{y}+1+x \\ \hline x\overline{y}^2+1+x+y \\ \overline{xy}+x\overline{y}+\overline{x}+x+\overline{x}^2y+y+xy+y^2 \end{array}\right]. \tag{C.5}$$

The automorphism $A^{d=7} = A_1A_{11}A_5A_{14}A_9$ with distance $d = 7$ is

$$A^{d=7} = \left[\begin{array}{cc|cc} \overline{x}+1 & \overline{x} & y & \overline{x}y+y \\ \overline{xy}+\overline{y}+1 & \overline{xy}+1 & x\overline{y}+1 & x\overline{y}+\overline{x}+1 \\ \hline \overline{xy}+\overline{y}+1 & \overline{xy}+1 & x\overline{y}+xy & x\overline{y}+\overline{x}+y+xy \\ \overline{x}+1 & \overline{x} & x+y & 1+x+\overline{x}y+y \end{array}\right]. \tag{C.6}$$

By applying $A^{d=7}$ on logical operatrs U_1, U_2, W, and stabilizer G, we can write down their polynomial representations as follows:

$$A^{d=7}U_1 = \left[\begin{array}{c} 0 \\ x\overline{y}^2+1 \\ \hline x\overline{y}^2+\overline{y}+x \\ \overline{y}+x\overline{y} \end{array}\right], \qquad A^{d=7}U_2 = \left[\begin{array}{c} \overline{x}+\overline{x}y \\ \overline{xy}+\overline{y}+\overline{x}+1 \\ \hline \overline{xy}+\overline{y}+1+y \\ \overline{x}+1+\overline{x}y \end{array}\right], \tag{C.7}$$

$$A^{d=7}W = \left[\begin{array}{c} \overline{x}y+y+xy+y^2 \\ x^2\overline{y}+\overline{x}+1+y \\ \hline x^2\overline{y}+\overline{x}+1+x+y+xy+x^2y+xy^2 \\ 1+x+x^2+\overline{x}y+y+y^2 \end{array}\right], \tag{C.8}$$

$$A^{d=7}G = \begin{bmatrix} \overline{xy}+\overline{x}^2+\overline{x}+1+\overline{x}y+y+xy+xy^2 \\ \overline{xy}^2+\overline{x}^2\overline{y}+\overline{xy}+x\overline{y}+1+y \\ \overline{xy}^2+\overline{x}^2\overline{y}+\overline{xy}+x^2\overline{y} \\ +1+x+y+xy+x^2y+xy^2 \\ \overline{xy}+\overline{x}^2+\overline{x}+x+x^2+\overline{x}y+y+y^2 \end{bmatrix}. \tag{C.9}$$

References

[1] D. Gottesman, *Stabilizer codes and quantum error correction*, (arXiv preprint) doi:10.48550/arXiv.quant-ph/9705052.

[2] P. Panteleev and G. Kalachev, *Quantum LDPC codes with almost linear minimum distance*, IEEE Trans. Inf. Theory **68**, 213 (2022), doi:10.1109/TIT.2021.3119384.

[3] P. Panteleev and G. Kalachev, *Asymptotically good Quantum and locally testable classical LDPC codes*, in *Proceedings of the 54th annual ACM SIGACT symposium on theory of computing*, Association for Computing Machinery, New York, USA, ISBN 9781450392648 (2022), doi:10.1145/3519935.3520017.

[4] T.-C. Lin and M.-H. Hsieh, *Good quantum LDPC codes with linear time decoder from lossless expanders*, (arXiv preprint) doi:10.48550/arXiv.2203.03581.

[5] I. Dinur, M.-H. Hsieh, T.-C. Lin and T. Vidick, *Good quantum LDPC codes with linear time decoders*, in *Proceedings of the 55th annual ACM symposium on theory of computing*, Association for Computing Machinery, New York, USA, ISBN 9781450399135 (2023), doi:10.1145/3564246.3585101.

[6] P. Jordan and E. Wigner, *Über das Paulische Äquivalenzverbot*, Z. Phys. **47**, 631 (1928), doi:10.1007/BF01331938.

[7] E. Fradkin, *Jordan-Wigner transformation for quantum-spin systems in two dimensions and fractional statistics*, Phys. Rev. Lett. **63**, 322 (1989), doi:10.1103/PhysRevLett.63.322.

[8] S. B. Bravyi and A. Y. Kitaev, *Fermionic quantum computation*, Ann. Phys. **298**, 210 (2002), doi:10.1006/aphy.2002.6254.

[9] R. Barends et al., *Digital quantum simulation of fermionic models with a superconducting circuit*, Nat. Commun. **6**, 7654 (2015), doi:10.1038/ncomms8654.

[10] T. Hensgens, T. Fujita, L. Janssen, X. Li, C. J. Van Diepen, C. Reichl, W. Wegscheider, S. Das Sarma and L. M. K. Vandersypen, *Quantum simulation of a Fermi-Hubbard model using a semiconductor quantum dot array*, Nature **548**, 70 (2017), doi:10.1038/nature23022.

[11] A. Mazurenko et al., *A cold-atom Fermi-Hubbard antiferromagnet*, Nature **545**, 462 (2017), doi:10.1038/nature22362.

[12] L. Tarruell and L. Sanchez-Palencia, *Quantum simulation of the Hubbard model with ultracold fermions in optical lattices*, Comptes Rendus Physique, Elsevier Masson (2018).

[13] F. Arute et al., *Observation of separated dynamics of charge and spin in the Fermi-Hubbard model*, (arXiv preprint) doi:10.48550/arXiv.2010.07965.

[14] B. H. Madhusudhana, S. Scherg, T. Kohlert, I. Bloch and M. Aidelsburger, *Benchmarking a novel efficient numerical method for localized 1d Fermi-Hubbard systems on a quantum simulator*, PRX Quantum **2**, 040325 (2021), doi:10.1103/PRXQuantum.2.040325.

[15] A. Y. Kitaev, *Fault-tolerant quantum computation by anyons*, Ann. Phys. **303**, 2 (2003), doi:10.1016/s0003-4916(02)00018-0.

[16] A. G. Fowler, M. Mariantoni, J. M. Martinis and A. N. Cleland, *Surface codes: Towards practical large-scale quantum computation*, Phys. Rev. A **86**, 032324 (2012), doi:10.1103/PhysRevA.86.032324.

[17] K. Setia, S. Bravyi, A. Mezzacapo and J. D. Whitfield, *Superfast encodings for fermionic quantum simulation*, Phys. Rev. Res. **1**, 033033 (2019), doi:10.1103/PhysRevResearch.1.033033.

[18] R. C. Ball, *Fermions without fermion fields*, Phys. Rev. Lett. **95**, 176407 (2005), doi:10.1103/PhysRevLett.95.176407.

[19] F. Verstraete and J. I. Cirac, *Mapping local Hamiltonians of fermions to local Hamiltonians of spins*, J. Stat. Mech.: Theory Exp. P09012 (2005), doi:10.1088/1742-5468/2005/09/p09012.

[20] J. D. Whitfield, V. Havlíček and M. Troyer, *Local spin operators for fermion simulations*, Phys. Rev. A **94**, 030301 (2016), doi:10.1103/PhysRevA.94.030301.

[21] M. Steudtner and S. Wehner, *Quantum codes for quantum simulation of fermions on a square lattice of qubits*, Phys. Rev. A **99**, 022308 (2019), doi:10.1103/PhysRevA.99.022308.

[22] C. Derby, J. Klassen, J. Bausch and T. Cubitt, *Compact fermion to qubit mappings*, Phys. Rev. B **104**, 035118 (2021), doi:10.1103/PhysRevB.104.035118.

[23] H. C. Po, *Symmetric Jordan-Wigner transformation in higher dimensions*, (arXiv preprint) doi:10.48550/arXiv.2107.10842.

[24] Y.-A. Chen, A. Kapustin and Đ. Radičević, *Exact bosonization in two spatial dimensions and a new class of lattice gauge theories*, Ann. Phys. **393**, 234 (2018), doi:10.1016/j.aop.2018.03.024.

[25] Y.-A. Chen and A. Kapustin, *Bosonization in three spatial dimensions and a 2-form gauge theory*, Phys. Rev. B **100**, 245127 (2019), doi:10.1103/PhysRevB.100.245127.

[26] Y.-A. Chen, *Exact bosonization in arbitrary dimensions*, Phys. Rev. Res. **2**, 033527 (2020), doi:10.1103/PhysRevResearch.2.033527.

[27] J. Wosiek, *A local representation for fermions on a lattice*, PhD thesis, Uniwersytet Jagielloński, Kraków, Poland (1981).

[28] A. Bochniak, B. Ruba, J. Wosiek and A. Wyrzykowski, *Constraints of kinematic bosonization in two and higher dimensions*, Phys. Rev. D **102**, 114502 (2020), doi:10.1103/PhysRevD.102.114502.

[29] A. Bochniak and B. Ruba, *Bosonization based on Clifford algebras and its gauge theoretic interpretation*, J. High Energy Phys. **12**, 118 (2020), doi:10.1007/JHEP12(2020)118.

[30] A. Bochniak, B. Ruba and J. Wosiek, *Bosonization of Majorana modes and edge states*, Phys. Rev. B **105**, 155105 (2022), doi:10.1103/PhysRevB.105.155105.

[31] H. Bombin, *Topological order with a twist: Ising anyons from an Abelian model*, Phys. Rev. Lett. **105**, 030403 (2010), doi:10.1103/PhysRevLett.105.030403.

[32] M. B. Hastings and A. Geller, *Reduced space-time and time costs using dislocation codes and arbitrary ancillas*, Quantum Inf. Comput. **15**, 962-986 (2015), doi:10.26421/QIC15.11-12-6.

[33] B. J. Brown, K. Laubscher, M. S. Kesselring and J. R. Wootton, *Poking holes and cutting corners to achieve Clifford gates with the surface code*, Phys. Rev. X **7**, 021029 (2017), doi:10.1103/PhysRevX.7.021029.

[34] T. I. Andersen et al., *Non-Abelian braiding of graph vertices in a superconducting processor*, Nature **618**, 264 (2023), doi:10.1038/s41586-023-05954-4.

[35] M. Steudtner and S. Wehner, *Fermion-to-qubit mappings with varying resource requirements for quantum simulation*, New J. Phys. **20**, 063010 (2018), doi:10.1088/1367-2630/aac54f.

[36] M. Steudtner and S. Wehner, *Quantum codes for quantum simulation of fermions on a square lattice of qubits*, Phys. Rev. A **99**, 022308 (2019), doi:10.1103/PhysRevA.99.022308.

[37] R. W. Chien and J. D. Whitfield, *Custom fermionic codes for quantum simulation*, (arXiv preprint) doi:10.48550/arXiv.2009.11860.

[38] Z. Jiang, A. Kalev, W. Mruczkiewicz and H. Neven, *Optimal fermion-to-qubit mapping via ternary trees with applications to reduced quantum states learning*, Quantum **4**, 276 (2020), doi:10.22331/q-2020-06-04-276.

[39] J. Bausch, T. Cubitt, C. Derby and J. Klassen, *Mitigating errors in local fermionic encodings*, (arXiv preprint) doi:10.48550/arXiv.2003.07125.

[40] M. Chiew and S. Strelchuk, *Discovering optimal fermion-qubit mappings through algorithmic enumeration*, Quantum **7**, 1145 (2023), doi:10.22331/q-2023-10-18-1145.

[41] C. Derby and J. Klassen, *A compact fermion to qubit mapping part 2: Alternative lattice geometries*, (arXiv preprint) doi:10.48550/arXiv.2101.10735.

[42] A. J. Landahl and B. C. A. Morrison, *Logical fermions for fault-tolerant quantum simulation*, (arXiv preprint) doi:10.48550/arXiv.2110.10280.

[43] B. Harrison, D. Nelson, D. Adamiak and J. Whitfield, *Reducing the qubit requirement of Jordan-Wigner encodings of N-mode, K-fermion systems from N to* $\lceil \log_2 \binom{N}{K} \rceil$, (arXiv preprint) doi:10.48550/arXiv.2211.04501.

[44] O. O'Brien and S. Strelchuk, *Ultrafast hybrid fermion-to-qubit mapping*, (arXiv preprint) doi:10.48550/arXiv.2211.16389.

[45] W. Cao, M. Yamazaki and Y. Zheng, *Boson-fermion duality with subsystem symmetry*, Phys. Rev. B **106**, 075150 (2022), doi:10.1103/PhysRevB.106.075150.

[46] Q.-S. Li, H.-Y. Liu, Q. Wang, Y.-C. Wu and G.-P. Guo, *A unified framework of transformations based on the Jordan-Wigner transformation*, J. Chem. Phys. **157**, 134104 (2022), doi:10.1063/5.0107546.

[47] M. G. Algaba, P. V. Sriluckshmy, M. Leib and F. Šimkovic IV, *Low-depth simulations of fermionic systems on square-grid quantum hardware*, (arXiv preprint) doi:10.48550/arXiv.2302.01862.

[48] Y.-A. Chen and Y. Xu, *Equivalence between fermion-to-qubit mappings in two spatial dimensions*, PRX Quantum **4**, 010326 (2023), doi:10.1103/PRXQuantum.4.010326.

[49] A. Kitaev, *Anyons in an exactly solved model and beyond*, Ann. Phys. **321**, 2 (2006), doi:10.1016/j.aop.2005.10.005.

[50] X.-G. Wen, *Quantum orders in an exact soluble model*, Phys. Rev. Lett. **90**, 016803 (2003), doi:10.1103/PhysRevLett.90.016803.

[51] C. Chen, P. Rao and I. Sodemann, *Berry phases of vison transport in $\mathbb{Z}_2$ topologically ordered states from exact fermion-flux lattice dualities*, Phys. Rev. Res. **4**, 043003 (2022), doi:10.1103/PhysRevResearch.4.043003.

[52] J. Nys and G. Carleo, *Variational solutions to fermion-to-qubit mappings in two spatial dimensions*, Quantum **6**, 833 (2022), doi:10.22331/q-2022-10-13-833.

[53] J. Nys and G. Carleo, *Quantum circuits for solving local fermion-to-qubit mappings*, Quantum **7**, 930 (2023).

[54] A. Rajput, A. Roggero and N. Wiebe, *Quantum error correction with gauge symmetries*, npj Quantum Inf. **9**, 41 (2023), doi:10.1038/s41534-023-00706-8.

[55] Z. Jiang, J. McClean, R. Babbush and H. Neven, *Majorana loop stabilizer codes for error mitigation in fermionic quantum simulations*, Phys. Rev. Appl. **12**, 064041 (2019), doi:10.1103/PhysRevApplied.12.064041.

[56] J. Haah, *Commuting Pauli Hamiltonians as maps between free modules*, Commun. Math. Phys. **324**, 351 (2013), doi:10.1007/s00220-013-1810-2.

[57] J. Haah, *Algebraic methods for quantum codes on lattices*, Rev. Colomb. Mat. **50**, 299 (2017), doi:10.15446/recolma.v50n2.62214.

[58] B. Ruba and B. Yang, *Homological invariants of Pauli stabilizer codes*, (arXiv preprint) doi:10.48550/arXiv.2204.06023.

[59] N. Tantivasadakarn, *Jordan-Wigner dualities for translation-invariant Hamiltonians in any dimension: Emergent fermions in fracton topological order*, Phys. Rev. Res. **2**, 023353 (2020), doi:10.1103/PhysRevResearch.2.023353.

[60] J. Haah, *Clifford quantum cellular automata: Trivial group in 2D and Witt group in 3D*, J. Math. Phys. **62**, 092202 (2021), doi:10.1063/5.0022185.

[61] D. Gottesman, *The Heisenberg representation of quantum computers*, (arXiv preprint) doi:10.48550/arXiv.quant-ph/9807006.

[62] S. Aaronson and D. Gottesman, *Improved simulation of stabilizer circuits*, Phys. Rev. A **70**, 052328 (2004), doi:10.1103/PhysRevA.70.052328.

[63] J. Haah, *Bifurcation in entanglement renormalization group flow of a gapped spin model*, Phys. Rev. B **89**, 075119 (2014), doi:10.1103/PhysRevB.89.075119.

[64] S. Vijay, J. Haah and L. Fu, *A new kind of topological quantum order: A dimensional hierarchy of quasiparticles built from stationary excitations*, Phys. Rev. B **92**, 235136 (2015), doi:10.1103/PhysRevB.92.235136.

[65] J. Haah, *Classification of translation invariant topological Pauli stabilizer codes for prime dimensional qudits on two-dimensional lattices*, J. Math. Phys. **62**, 012201 (2021), doi:10.1063/5.0021068.

[66] A. Dua, I. H. Kim, M. Cheng and D. J. Williamson, *Sorting topological stabilizer models in three dimensions*, Phys. Rev. B **100**, 155137 (2019), doi:10.1103/PhysRevB.100.155137.

[67] R. W. Chien and J. Klassen, *Optimizing fermionic encodings for both Hamiltonian and hardware*, (arXiv preprint) doi:10.48550/arXiv.2210.05652.

[68] R. W. Chien, K. Setia, X. Bonet-Monroig, M. Steudtner and J. D. Whitfield, *Simulating quantum error mitigation in fermionic encodings*, (arXiv preprint) doi:10.48550/arXiv.2303.02270.

[69] D. González-Cuadra et al., *Fermionic quantum processing with programmable neutral atom arrays*, Proc. Natl. Acad. Sci. **120**, e2304294120 (2023), doi:10.1073/pnas.2304294120.