

Tailoring Dynamical Codes for Biased Noise: The X³Z³ Floquet Code

F. Setiawan^{1,*} and Campbell McLauchlan^{2,3,†}

¹Riverlane Research Inc., Cambridge, Massachusetts 02142, USA

²Centre for Engineered Quantum Systems, School of Physics,
The University of Sydney, Sydney, NSW 2006, Australia

³Riverlane, Cambridge, CB2 3BZ, UK

We propose the X³Z³ Floquet code, a dynamical code with improved performance under biased noise compared to other Floquet codes. The enhanced performance is attributed to a simplified decoding problem resulting from a persistent stabiliser-product symmetry, which surprisingly exists in a code without constant stabilisers. Even if such a symmetry is allowed, we prove that general dynamical codes with two-qubit parity measurements cannot admit one-dimensional decoding graphs, a key feature responsible for the high performance of bias-tailored stabiliser codes. Despite this, our comprehensive simulations show that the symmetry of the X³Z³ Floquet code renders its performance under biased noise far better than several leading Floquet codes. To maintain high-performance implementation in hardware without native two-qubit parity measurements, we introduce ancilla-assisted bias-preserving parity measurement circuits. Our work establishes the X³Z³ code as a prime quantum error-correcting code, particularly for devices with reduced connectivity, such as the honeycomb and heavy-hexagonal architectures.

I. INTRODUCTION

Quantum error correction (QEC) [1–4] should be understood as occurring both in space and time [5]. Taking advantage of the temporal dimension, Floquet codes [6–20], or more generally dynamical codes [21–23], form a large class of quantum error-correcting codes, which can achieve competitive fault-tolerant performance [6, 8–17] while reducing the weights of check measurements performed for the error correction. Several of these codes [7, 9, 13, 18] also benefit from being defined on a lattice with sparser connectivity than that for the surface code [24, 25]: each qubit is only connected to three other qubits. In architectures where two-qubit parity check measurements are native, Floquet codes could achieve higher thresholds [8, 10, 17] and lower space-time overheads [8, 10] than the surface code. Without requiring additional connectivity, such codes can be deformed around defective components caused by highly noisy qubits or gates [15, 26]. Moreover, dynamical codes allow for implementations of arbitrary Clifford, and even some non-Clifford gates, through low-weight parity check measurements [21]. In addition, dynamical measurement schedules can result in certain errors being self-corrected in the Floquet-Bacon-Shor code [20] – this code was recently demonstrated in superconducting qubit experiments [27].

Although Floquet codes (in particular, the honeycomb code) have been studied under various noise models [9, 28], there have not been any Floquet codes that are specifically tailored for an improved performance under biased noise. A biased noise model is one in which a specific type of error, for example, phase errors, occurs more

frequently than other errors, such as bit flip errors. This biased noise is typical to most quantum platforms, for example, bosonic “cat” qubits [29, 30], spin-optical [17, 31], neutral atoms [32], quantum-dot spin qubits [33, 34] and Majorana qubits [35, 36]. In these platforms, biased noise can arise due to different predominant error mechanisms. For example, by increasing the photon number of the resonators, the cat qubits [29, 30] can be made exponentially protected against the bit-flip error at the expense of only a linear increase in the phase-flip error. For the spin qubits in both spin-optical [17] and quantum dot architectures [33, 34], the noise is predominantly dephasing due to the short spin coherence time, T_2 . Moreover, the two-qubit gates of quantum-dot spin qubits have also been experimentally shown to exhibit phase-biased noise which is caused by the T_2 dephasing, non-Markovian error sources, AC-Stark shift, and calibration errors [34]. In neutral-atom qubits, detailed modeling of the experiment shows that the two-qubit gate noise is predominantly phase-biased due to the short Rydberg state T_2^* lifetime caused by finite atomic temperature and light-shift fluctuations of the lasers [32]. Lastly, for Majorana qubits, the noise is expected to be biased due to residual coupling between the Majoranas and thermal fluctuations that excite the qubit to an above-gap quasiparticle state [37].

For enhanced performance, quantum-error-correcting codes need to be designed such that they possess symmetries that can be utilised to simplify the error-syndrome decoding problem given the noise structure [38–47]. While there have been a number of proposals for bias-tailored static codes [39–55], designing Floquet codes for high performance under biased noise is still an open problem. Owing to the experimental relevance of biased noise and given the ease of implementation of Floquet codes, which require only two-qubit parity measurements that can be implemented even in architectures with sparse connectivity, it is therefore imperative to tailor Floquet

* setiawan.wenming@gmail.com

† campbell.mclauchlan@gmail.com

codes for biased noise and study how the performance of such dynamical codes can be improved.

In this paper, we present the X^3Z^3 bias-tailored Floquet code, a Clifford-deformed [46] version of the Calderbank–Shor–Steane (CSS) Floquet code [13, 18]. Despite not having a fixed stabiliser group (as static codes have), remarkably the X^3Z^3 Floquet code possesses a stabiliser-product symmetry under infinitely phase-biased noise, simplifying decoding in biased noise regimes. We perform an in-depth study of this code, along with the CSS Floquet code [13, 18], and two types of honeycomb codes: one introduced by Hastings and Haah [6] and the other by Gidney *et al.* [8]. We simulate all codes under biased-noise models, and find that the X^3Z^3 code has the best performance. Using a matching decoder, we find that, as the noise changes from fully depolarising to pure dephasing, the X^3Z^3 Floquet code threshold increases from 1.13% to 3.09% under a code-capacity noise model and increases from 0.76% to 1.08% under a circuit-level noise model mimicking hardware with noisy direct entangling measurements. Furthermore, we show that its sub-threshold performance is also substantially better under biased noise than other Floquet codes.

Compared to its static counterparts, the X^3Z^3 Floquet code has an advantage that it can be realised using only two-qubit parity check measurements. This makes it particularly suitable for devices with constrained connectivity, such as the honeycomb and heavy-hexagonal lattice (currently IBM’s preferred superconducting-qubit architecture) [56, 57]. Moreover, we demonstrate that the two-qubit parity measurements of the Floquet code can be performed in a bias-preserving way even in hardware without direct entangling measurements, thus enabling high performance implementation in such devices. Crucially, we show that even in the presence of mid-gate errors which degrade noise bias in the target qubits of conventional CNOT gates [58], two of the three required parity-check circuits, built using these conventional gates, can be made fully Z -bias preserving, i.e., sustain the Z noise bias on both data qubits, while the other one is partially phase-bias preserving, i.e., protects the Z noise bias in only one of the data qubits.

We argue that other dynamical codes defined on the same architecture, and built from two-body measurements, would likely not have drastically improved performance compared to the X^3Z^3 Floquet code. To support this argument, we prove that decoding graphs of such dynamical codes under infinitely phase-biased noise have connectivities that are too high for the decoding problem to be reduced to a simple decoding of repetition codes, as is the case for static codes [42, 46]. This can be understood as resulting from the fact that error syndromes of dynamical codes possess less symmetry than their static code counterparts.

The paper is laid out as follows. In the Secs. II A–II C, we first review the basics of Floquet codes together with two widely studied examples: honeycomb and CSS Floquet codes. Readers who are already familiar with

Floquet codes can skip directly to Sec. II D in which we discuss our X^3Z^3 Floquet code. In Sec. II E, we show that there exists a persistent symmetry in the code’s error syndrome under pure dephasing noise that allows for a simplified decoding. Subsequently, in Sec. II F we introduce ancilla-assisted bias-preserving parity measurement circuits that allow for high-performance code implementation in devices without native entangling measurements. In Secs. II G and II H, we provide the simulation results for memory experiments. In Sec. III, we give two theorems showing that the decoding hypergraphs of dynamical codes cannot be reduced to 1D graphs. Finally, we conclude and present future research directions in Sec. III. In Sec. IV, we provide the description of our noise models, details of numerical simulations and proofs of the no-go theorems. In the Supplementary Information, we provide a more thorough review of the basics of honeycomb and CSS Floquet codes, details of our parity check circuits, plots for determining thresholds, details of hyperedges in the honeycomb codes, and results for Floquet codes with elongated dimension and twisted periodic boundary conditions.

II. RESULTS

A. Dynamical and Floquet codes

We begin by defining dynamical and Floquet codes. Here we consider the Floquet codes to be defined on the lattice of a two-dimensional colour code, which is trivalent and three-colourable. A trivalent lattice has each vertex incident to three edges, and a three-colourable lattice has every face assigned one of three colours in such a way that there are no two adjacent faces of the same colour. Throughout this paper, we will use the honeycomb lattice as an example of such a lattice (see Fig. 1).

We define qubits as residing on the vertices of the lattice and error-correction is performed by measuring two-qubit operators defined on edges of the lattice (i.e., acting on qubits incident to a given edge). Each edge is coloured the same as the plaquettes it connects [see e.g., the coloured edges in the middle right of the lattice in Fig. 1(a)]. We perform error-correction by measuring edge operators in a particular sequence. After any given subround of edge measurements, the system will be in the +1-eigenstate of the operators in an “instantaneous stabiliser group” (ISG), which will change at each time step. The ISG at time step $t+1$ is defined as

$$\text{ISG}_{t+1} = \langle S, \pm M \mid M \in \mathcal{M}_{t+1}, S \in \text{ISG}_t, \\ \text{such that } [S, M] = 0, \forall M \in \mathcal{M}_{t+1} \rangle. \quad (1)$$

In the above, \mathcal{M}_{t+1} is the set of “check” measurements performed at time step $t+1$. The ISG_{t+1} also includes “plaquette stabiliser operators” $S \in \text{ISG}_t$ which commute with all $M \in \mathcal{M}_{t+1}$. The check measurement operators are chosen in such a way that those check operators at time t that have overlapping qubit supports

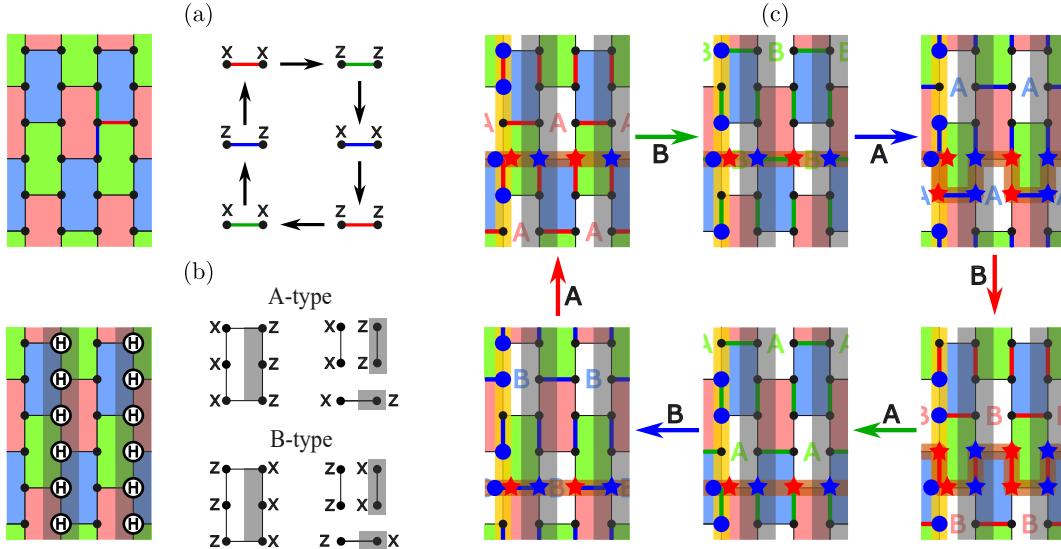


FIG. 1. X^3Z^3 and CSS Floquet codes on the hexagonal lattice with their stabilisers, checks, and logical operators. (a) CSS and (b,c) X^3Z^3 Floquet codes. (a,b) Left: Hexagonal lattice with qubits sitting on vertices and opposite boundaries identified. Plaquettes and edges are assigned one of three colours [red, green, blue; examples are shown in (a)] and one of two types (X- or Z-type for CSS and A- or B-type for X^3Z^3). (a) Right: CSS Floquet code measurement schedule. (b) Left: The X^3Z^3 Floquet code is obtained from the CSS code via Hadamard gates applied to shaded domains' qubits. Right: Plaquette and edge (check) operators are A- or B-type, depending on their support basis on the shaded/unshaded domains. (c) The X^3Z^3 Floquet code measurement schedule. Arrows indicate the type and colour of the edge operator measured at each step, where the edges just measured (members of the ISG) are highlighted in the lattice. Uncoloured plaquettes host only a single type of stabiliser, either A or B, indicated by the letters in the plaquettes, while coloured plaquettes host both A- and B-type stabilisers in the ISG. One set of anti-commuting logical operators is shown by yellow and brown strings, where their qubit supports are depicted using big circles and stars, respectively, with the X (Z) bases highlighted by red (blue) colouring. The other set (not shown) is similar to the set shown but offset by three measurement subrounds and with an $X \leftrightarrow Z$ interchange of the qubit supports' bases.

with check operators at $t + 1$ anti-commute. For Floquet codes, the measurement sequence is periodic, such that $\mathcal{M}_{t+T} = \mathcal{M}_t$ for some integer T . For such a code, we will be performing quantum memory experiments with mT time steps, for some integer m . We will refer to m as the number of “QEC rounds” in the experiment, while we will refer to mT as the number of “measurement sub-rounds” in the experiment.

The logical operators at time t are given by $\mathcal{C}(\text{ISG}_t)/\text{ISG}_t$, where $\mathcal{C}(\text{ISG}_t)$ is the centraliser of ISG_t , i.e., the group of Pauli operators commuting with all $S \in \text{ISG}_t$. A (potentially trivial) logical operator “representative” is some member of $\mathcal{C}(\text{ISG}_t)$. Each lowest-weight nontrivial logical operator representative for the codes considered will be a string-like Pauli operator at each time step [see Fig. 1(c)]. To avoid anti-commuting with the next-subround edge measurements, certain check measurement results along a logical operator’s path have to be multiplied into that logical operator. Hence, the logical operators will evolve from one time step to the next. As an illustration, take for example the vertical logical operator immediately after the gB checks are applied, i.e., the operator on the yellow string in the top-centre panel of Fig. 1(c). This operator is obtained by multiplying the vertical green-B checks on the vertical

logical operator’s path [the two green edges on the yellow path shown in the top-centre panel of Fig. 1(c)] into the previous time-step vertical logical operator [the operator shown by the yellow string in the top-left panel of Fig. 1(c)]. Note that after the multiplication, the logical operator commutes with the next check measurements, which are the blue-A edge measurements.

We can detect errors if we can find sets of measurements, called detectors, that always multiply to the value $+1$ in the absence of noise, thus registering no error. Over some number of QEC rounds we will have extracted several detector outcomes. A detector (or decoding) hypergraph is formed by first defining a node for each (independent) detector in the code’s history. Subsequently, for each potential fault (e.g., Pauli or measurement errors) that might have occurred, a (hyper)edge is drawn between the detectors whose signs are flipped by this fault. Each (hyper)edge is assigned a weight based on the probability of the corresponding error occurring [59]. The codes we will be examining are amenable to minimum-weight perfect matching decoding [60], upon decomposing hyperedges into edges. Given a “syndrome” (a set of detectors whose measurements return -1 rather than $+1$), the decoder attempts to pair up the triggered detectors to determine a highest-probability correction operation.

The decoder succeeds if the error combined with the correction is a trivial logical operator.

Having discussed the general idea of Floquet codes, we now briefly review the commonly studied examples: two variants of the honeycomb code [6, 8] and the CSS Floquet code [13, 18]. We will later modify the CSS Floquet code to achieve the bias-tailored X^3Z^3 Floquet code. More details of the honeycomb and CSS Floquet codes are presented in Supplementary Sec. I of the Supplementary Information [61].

B. Honeycomb codes

We begin by first discussing the honeycomb codes. The first variant we review is due to Gidney *et al.* [8], which we call the P^6 Floquet code, since its plaquette operators are six-body operators of the form $P^{\otimes 6}$ for $P = X, Y, Z$. We define edge operators of three types: on red edges we define an XX operator, on green edges a YY operator and on blue edges a ZZ operator. We measure edge operators in the periodic sequence $r \rightarrow g \rightarrow b$. When this code is defined with periodic boundary conditions it stores two logical qubits (it is equivalent to the toric code concatenated with a two-qubit repetition code at each time step [6]). The code's logical operators evolve through the measurement cycle (see Supplementary Sec. IA of the Supplementary Information [61]). While the measurement sequence has period 3, the logical operators only return to their initial values (up to signs) with period 6.

We define one stabiliser operator for each plaquette, where red, green, and blue plaquettes host the $X^{\otimes 6}$, $Y^{\otimes 6}$, and $Z^{\otimes 6}$ operators, respectively. These plaquette operator eigenvalues are inferred from edge measurements in two consecutive subrounds. Detectors are formed from consecutive plaquette operator measurements.

The second honeycomb code variant, which we call the XYZ^2 honeycomb code, is due to Hastings and Haah [6]. It differs from the P^6 code by single-qubit Clifford rotations acting on the qubits. While the XYZ^2 code's edges are still coloured the same as the P^6 edges, the XYZ^2 code's edge operators have their Pauli bases defined according to their orientations within each T junction, i.e., horizontal edges are of Z type while the X and Y checks are respectively those edges which are 90° clockwise and counter-clockwise rotated from the horizontal edges (see Supplementary Sec. IA of the Supplementary Information [61]). All plaquettes have the same stabiliser operator, i.e., the XYZ^2 operator. While the logical operators of the XYZ^2 code have the same qubit supports as those of the P^6 code, the qubit support bases of the XYZ^2 logical operators are not uniform throughout, but involve X , Y , and Z Paulis [6]. We note that even though these honeycomb code variants have been studied by several works in the literature (e.g., Refs. [6–9]), there have not been any

studies comparing the performance of these two codes under biased noise.

C. CSS Floquet code

Having discussed the honeycomb codes, we now give a brief review of another type of Floquet code, the CSS Floquet code, which was first presented in Refs. [13, 18]. We refer the readers to Supplementary Sec. IB of the Supplementary Information [61] for more details on the code. The CSS Floquet code is defined on the same honeycomb lattice. We show the code's measurement cycle in Fig. 1(a). We measure operators defined on edges using the $r \rightarrow g \rightarrow b$ cycle, but alternate between measuring XX and ZZ operators on these edges. Hence, this code has a period-6 measurement cycle, and its logical operator evolutions also have period-6 (see Supplementary Fig. 2 of the Supplementary Information [61] for an illustration). Even though the honeycomb code's measurement schedule has only a period of 3, for consistency, we will define 1 QEC round to be 6 measurement subrounds for all codes studied in this paper. The check operators, stabilisers and logicals all are either X - or Z -type and, for this reason, the code is called the CSS Floquet code.

Unlike the honeycomb codes, the CSS Floquet code has no persistent stabiliser operators. Instead, each plaquette stabiliser is repeatedly reinitialised and measured out, with subsequent measurement values compared to form a detector. In contrast to the honeycomb codes where the values of the plaquette stabilisers are inferred from measurements of check operators in two consecutive subrounds, here the plaquette operator measurements take only a single subround [13, 18]. Because of these single-step measurements, at each measurement subround, there is always one type of plaquette operator that anti-commutes with the check measurements, and hence their values are undetermined (see Supplementary Sec. IB of the Supplementary Information [61]). These “inactive” plaquettes, however, will be reinitialised at the next measurement subround. As a result, at each time step, the ISG contains X -type and Z -type stabiliser operators defined on two of the three colours of plaquette and *either* an X -type *or* a Z -type operator defined on the other colour. For instance, after measuring the red- X checks, the ISG contains $X^{\otimes 6}$ and $Z^{\otimes 6}$ operators on both blue and green plaquettes, but only $X^{\otimes 6}$ red plaquette operators, since $Z^{\otimes 6}$ red plaquette operators anti-commute with the red- X checks (see Supplementary Sec. IB of the Supplementary Information [61] for more details).

The CSS Floquet code is naturally suited to a minimum-weight perfect matching (MWPM) decoding, since single-qubit (X or Z) Pauli and measurement errors all lead to graph-like syndromes [13]: they each trigger a pair of detectors. There are two decoding graphs formed from the Z -type and X -type detectors.

Only Y errors form hyperedges that need to be decomposed into edges in the two detector graphs.

D. X^3Z^3 bias-tailored Floquet code

We now describe the X^3Z^3 bias-tailored Floquet code, which is shown in Figs. 1(b) and 1(c). This code is related to the CSS Floquet code by Hadamard gates applied to the qubits in alternating strips [i.e., the grey strips in Fig. 1(b)] along vertical non-trivial cycles of the lattice, and is a Floquetified version of the domain wall colour code [46]. As in the CSS Floquet code, the X^3Z^3 code also has two types of plaquettes and edges: one originates from the Pauli X and the other from the Pauli Z plaquettes and edges in the CSS code before the Hadamard deformation. We refer to these modified operators as A-type and B-type, respectively; these are shown in Fig. 1(b). The measurement sequence is analogous to the CSS Floquet code sequence: $rA \rightarrow gB \rightarrow bA \rightarrow rB \rightarrow gA \rightarrow bB$, where cA represents the measurement of A-type check operators along c -coloured edges, and similarly for cB . Just as with the CSS Floquet code [13], this code can also be defined on a planar lattice with boundary.

The decoding of the X^3Z^3 Floquet code under Z -biased noise is simplified by the presence of a symmetry in the decoding graphs, which we will explain below. This is a space-time analogue of the symmetries present in bias-tailored static codes such as the XZZX code [42] and the domain wall colour code [46]. In such codes, under the infinitely phase-biased code-capacity noise model (with only single-qubit Z errors, for example), syndromes are forced to come in pairs along one-dimensional (1D) strips of the lattice. This results from strips of stabilisers multiplying together to the identity along one domain and commuting with all Z errors on all qubits: these are 1D symmetries of the stabiliser code under infinitely phase-biased noise. We will refer to (plaquette) stabilisers in the ISG that are flipped by an error as anyons. In bias-tailored static codes, anyons can propagate within strips but cannot move outside the strip without changing their Pauli type. We will show that there also exists a similar symmetry in the bias-tailored Floquet code.

As in the CSS Floquet code, there are two disjoint decoding graphs for the X^3Z^3 Floquet code: the A-type and B-type graphs, whose edges correspond to single-qubit Pauli errors. In even measurement subrounds, we perform B-type measurements and form detectors for the B-type graph while in odd measurement subrounds we only form detectors for the A-type graph. Note that under a more complicated noise model, such as one that includes measurement or two-qubit errors, edges can exist between the A- and B-type graphs and they are no longer disjoint.

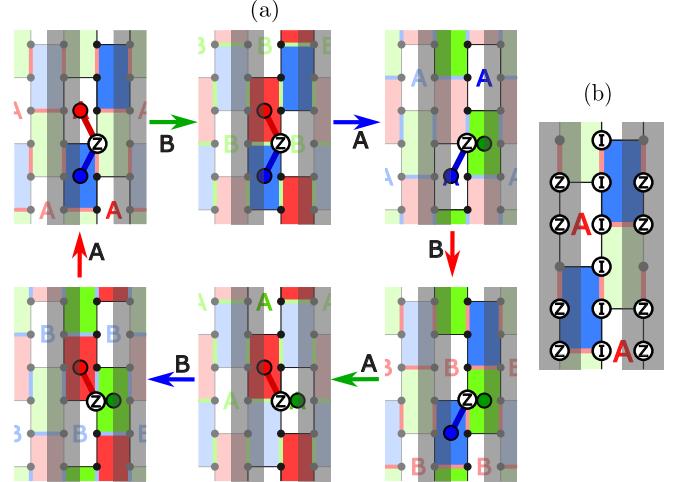


FIG. 2. Persistent stabiliser-product symmetry of the X^3Z^3 Floquet code leads to improved performance under Z -biased noise. (a) An example of such a symmetry along a single unshaded strip of the lattice, together with a Pauli- Z error, is shown throughout the measurement cycle. Arrows indicate the type of check (cA or cB for some colour c) measured at each subround. A-type plaquette operators whose product forms the symmetry at that subround are highlighted in darker colours. Plaquettes indicated by coloured dots are the locations of syndromes that would be triggered if the Z error shown were to occur at that measurement subround. (b) An example of a stabiliser-product symmetry just before gB checks are measured is the product of red and blue A-type plaquettes shown. Since this product is the identity along the vertical unshaded strip and commutes with Z errors on all qubits, the syndromes/anyons [coloured dots in (a)] must appear in pairs along the vertical strips.

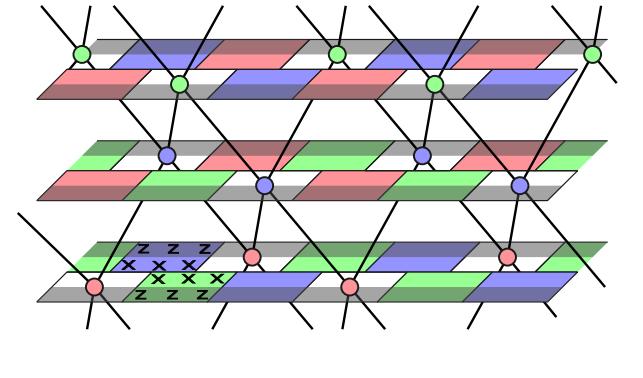


FIG. 3. The 2D A-type detector graph of the X^3Z^3 Floquet code under pure dephasing noise. Only a single vertical strip [e.g., that supporting the stabiliser-product symmetry in Fig. 2(b)] is shown. The illustrated part of the graph is disconnected from those of neighbouring strips. Arrows represent only the A-type measurements. The B-type subrounds are not shown, since they do not influence the detectors in this graph. Each node (representing a detector) is placed in the centre of a plaquette and at the subround in which the corresponding detector is formed. For example, the bottom layer corresponds to the blue A-check measurements, at which time the red A-type plaquettes are measured.

E. Persistent stabiliser-product symmetry and two-dimensional decoding of the X^3Z^3 Floquet code under biased noise

There is no constant stabiliser group when viewing the X^3Z^3 Floquet code as a subsystem code (in a subsystem code, the stabiliser group is defined as the centre of the gauge group, which, in our case, is generated by all edge operators) [13, 18]. Surprisingly, despite this fact, at every time step there do exist operators that form a symmetry under pure dephasing noise (without measurement errors). An example of such a symmetry is shown in Fig. 2. It is a symmetry on one of the unshaded domains and it is formed by the product of A-type plaquettes. A similar symmetry can also be found on the shaded strips which are formed by the product of B-type plaquettes.

To see why such symmetries exist, consider one particular time step, e.g., just before the green B-type measurements, and one vertical unshaded domain in the lattice. This is shown in Fig. 2(b). As depicted, the product of red and blue A-type plaquettes in the ISG is the identity on all qubits in the domain, and yields Pauli Z operators on some qubits in the neighbouring shaded vertical domains. To be more concrete, consider the plaquettes that can detect Z errors along the unshaded vertical domain \mathcal{D} at time step t . These are the A-type plaquette operators with support in \mathcal{D} . For a plaquette stabiliser operator S , which is measured in a time step $t' > t$, to form a detector (i.e., a node in the detector graph), it has to be in ISG_t and ISG_{t+1} , since it must commute with at least the next measurement subround. We can define $\text{ISG}_{(t,t+1)}^{A,\mathcal{D}}$ as the subgroup of $\text{ISG}_t \cap \text{ISG}_{t+1}$ formed from A-type plaquettes with non-trivial supports on domain \mathcal{D} . We then have

$$\prod_{S \in \text{ISG}_{(t,t+1)}^{A,\mathcal{D}}} S = \left(\bigotimes_{k \in \mathcal{D}} I_k \right) \left(\bigotimes_{j \in \bar{\mathcal{D}}_{Z,t}} Z_j \bigotimes_{j' \in \bar{\mathcal{D}}_{I,t}} I_{j'} \right), \quad (2)$$

where $\bar{\mathcal{D}}_{Z,t}$ and $\bar{\mathcal{D}}_{I,t}$ are some subsets of qubits in the domains adjacent to domain \mathcal{D} on which the product of the A-type plaquette stabilisers yields Pauli Z and the identity, respectively [for the example given in Fig. 2(b)], the domain \mathcal{D} is the unshaded vertical strip, the domains $\bar{\mathcal{D}}_{Z,t}$ and $\bar{\mathcal{D}}_{I,t}$ comprise the qubits in the neighbouring shaded vertical strips with Z -circled label and no label, respectively]. We use a subscript t in the notations $\bar{\mathcal{D}}_{Z,t}$ and $\bar{\mathcal{D}}_{I,t}$ to indicate that the qubits belonging to these two subsets depend on the time step t . Note that we suppress the identity factors outside of \mathcal{D} and its adjacent domains. In particular, by restricting this operator to domain \mathcal{D} , we have the following persistent stabiliser-product symmetry:

$$\prod_{S \in \text{ISG}_{(t,t+1)}^{A,\mathcal{D}}} S|_{\mathcal{D}} = \bigotimes_{k \in \mathcal{D}} I_k, \quad (3)$$

which implies that the error syndrome obeys a conservation law in the domain \mathcal{D} : the syndrome comes in

pairs along this strip. At each subround, there also exists a similar symmetry on the shaded domains, which is formed by the product of B-type plaquettes. Therefore, for the pure Z Pauli noise model, we can perform minimum-weight perfect matching decoding *within* each domain.

To explain this more concretely, we use the anyon picture. These anyons, shown in Fig. 2(a), are interpreted as having locations given by the detectors that *would* be flipped if the corresponding error were to occur at that time step. For example, if the Z error shown in Fig. 2(a) occurs just before the green B-check measurements (top-left panel of the figure), it will trigger a red A detector after two measurement subrounds and a blue A detector after a further two subrounds. As a result of each of the symmetries, Pauli- Z errors create anyons in pairs along each vertical strip. An example of such anyons, which are formed due to a Pauli Z error at a particular measurement subround, is shown in Fig. 2(a). While the plaquette locations of the anyons, triggered by the Z error occurring on the same qubit, change every other subround, they always respect the symmetry, i.e., are aligned along the domain (see Fig. 2).

Unlike the symmetry in static stabiliser codes, this stabiliser-product symmetry does not allow for 1D decoding even when measurement errors are absent. While in static codes, only measurement errors produce “time-like” edges, in Floquet codes, even Pauli errors produce time-like edges (between detectors formed at different times). To demonstrate this, we display in Fig. 3 a portion of the A-type detector graph under infinitely phase-biased single-qubit noise with no measurement errors. As can be seen, there is a disconnected subgraph defined along one unshaded domain of the code lattice. Even considering this simple noise model, the decoding graph in the infinite bias regime is two-dimensional (i.e., the graph is planar). This results from the fact that neighbouring plaquettes are measured at different times. We emphasise that although the above discussion is based on Z errors, the same analysis also holds for X errors because X and Z operators are interchangeable for the X^3Z^3 code. As a result, the performance of the X^3Z^3 code under X -biased noise is expected to be the same as that under the Z -biased noise model.

F. Bias-preserving parity-check circuits

While two-body measurements are native to certain architectures, for example, Majorana [10, 62, 63] and photonic [17, 64] platforms, most architectures require quantum circuits to carry out such measurements. To maintain the high-performance of bias-tailored codes in these hardware, the parity-check measurement circuits need to be constructed in a phase-bias preserving manner, such that they do not propagate frequently-occurring errors to rarely-occurring errors. To this end, we design

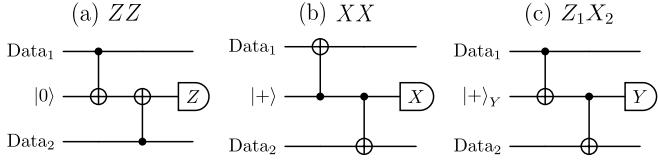


FIG. 4. Depth-2 parity check measurement circuits. For the circuits to be bias-preserving, the CNOT gates in the circuits need to be also bias-preserving. Shown are the circuits for (a) Z_1Z_2 , (b) X_1X_2 , and (c) Z_1X_2 checks circuits, where the ancilla qubit is reset and later measured in the Z , X and Y bases, respectively.

two-qubit parity check circuits that preserve the Z bias on data qubits. That is, the probability of X and Y errors on data qubits after these circuits remains small (proportional to single-qubit X or Y error probabilities). Our bias-preserving parity check measurement circuits are constructed by using an ancilla circuit connecting two data qubits, which can be realised even in devices with minimal connectivity, such as the heavy-hexagonal layout [56, 57].

For a circuit to be bias preserving, it can be constructed using only gates that do not change the error type under conjugation. For example, one way to construct bias-preserving circuits is to use only CNOT gates in the measurement circuits as they only propagate errors to others with the same Pauli type. For the X^3Z^3 code, we need to construct bias-preserving circuits for three different kinds of parity checks: ZZ , XX and ZX . All these three circuits can be constructed using only CNOT gates as shown in Fig. 4. As depicted, these circuits also include resets and measurements of the ancilla qubit in the three Pauli bases, i.e., Z , X and Y bases for ZZ , XX and ZX checks, respectively. Note that the circuits are still phase-bias preserving even if the resets/measurements in the X and Y bases are compiled in terms of resets/measurements in the Z basis with additional single-qubit Clifford gates. This is because a Z error on any qubit at any time step in the compiled circuits still propagates to data qubits as a Z error, or as a check operator about to be measured.

The parity-check circuits shown in Fig. 4, however, may not preserve the noise bias if errors happen during the CNOT gates. This is because a Z error occurring on the target qubit during the application of a conventional (non-bias preserving) CNOT gate may result in a combination of Z and X errors on the target qubit after the CNOT gate [58]. One way to remedy this problem is to use bias-preserving CNOT gates [58]. These gates, however, are available only in specific platforms such as cat qubits. Here, we propose a more general approach to constructing Z -bias preserving parity check circuits without requiring the bias-preserving gates. While our proposed circuits can preserve only a specific type of noise bias, i.e., the Z bias, these circuits are built using conventional (which may not be phase-bias preserving)

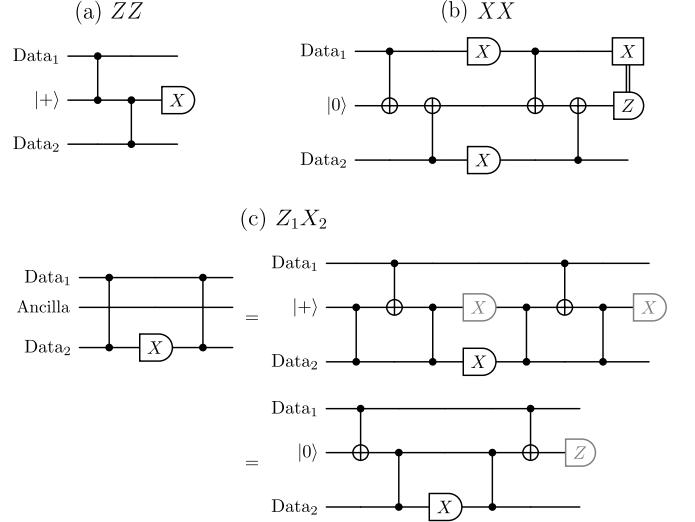


FIG. 5. Parity-check measurement circuits built from conventional two-qubit gates and constructed in ways that the Z -noise bias on the data qubits is preserved under mid-gate errors. Shown are (a) ZZ , (b) XX , and (c) Z_1X_2 check circuits. The circuits in (a) and (c) preserve the Z noise bias on data qubits even in the presence of mid-gate errors. The circuit in (b) includes a classically controlled X gate targeting one of the data qubits, which is non- Z bias preserving. Data qubits are labelled by Data₁ and Data₂, while the middle qubit in each circuit is a measurement ancilla. In (c), the first equality can be checked by noting that the depth-3 circuit before and after the mid-circuit measurement performs a next-nearest-neighbour CZ gate, if the measurement ancilla is initially in the $|+\rangle$ state. The second equality can be checked by commuting all gates past the measurement. The grey measurements can be optionally included to provide flag information.

two-qubit gates and thus can be implemented in many different architectures.

Our proposed phase-bias preserving parity-check circuits are depicted in Fig. 5. For the ZZ and ZX check measurement circuits, they are constructed using either CZ gates or CNOT gates targeting ancilla qubits, which propagate Pauli errors on the ancilla qubits only as Z errors on the data qubits. In other words, these circuits do not leave any residual Pauli X or Y errors occurring with probability $\mathcal{O}(p_Z)$ and therefore do not degrade the Z noise bias on the data qubits. This is true even if we consider mid-gate Pauli errors (see Supplementary Sec. II in the Supplementary Information [61] for details). The XX measurement circuit shown in Fig. 5(b), however, is not fully Z -bias preserving in the presence of mid-gate errors. This circuit degrades the Z bias on one of the data qubits, i.e., the data qubit that is the target of the classically-controlled X gate. (Note that instead of implementing the classically controlled X gate, we may alternatively perform a CNOT gate, with the control being the ancilla qubit and target being data qubit 1, immediately preceding the ancilla measurement.) However, since the XX parity-check circuit shown in Fig. 5(b) degrades

the Z -bias on only one data qubit, it has a better Z -bias-preserving performance compared to the XX check circuit in Fig. 4(b), which degrades the Z -bias on both data qubits due to the fact that both data qubits are used as target qubits of the CNOT gates in the circuit.

We now elaborate on how the parity-check circuits work. We begin by noting that while the ZZ parity check measurements [Fig. 5(a)] are performed by reading out the ancilla qubits like in standard syndrome extraction circuits, the XX and ZX parity values [Figs. 5(b) and 5(c)] are obtained by measuring the data qubits after some two-qubit gates are applied on them. These two-qubit gates are required for the parity measurements so as not to reveal each individual data qubit's state. For the XX parity-check circuit, the two mid-circuit X -measurements in Fig. 5(b), are equivalent to reading out the X_1X_a and X_2X_a parity values, where the Pauli operator X_a acts on the ancilla qubit which is prepared in the $|0\rangle$ state. Hence, the product of these two measurement outcomes is the required X_1X_2 parity check outcome. We can disentangle the ancilla qubit by performing a final Z_a measurement (which requires correcting a potential bit flip on one of the data qubits with the classically controlled X gate), or by using a CNOT gate (not shown) before the Z_a measurement (see Supplementary Sec. II of the Supplementary Information [61] for details).

The mixed-type Z_1X_2 parity circuit can be constructed using CZ gates between the data qubits, as shown in Fig. 5(c). If the connections between the data qubits are not available in the device such as in hardware with a heavy-hexagonal lattice, these CZ gates have to be implemented using ancilla qubits in-between the data qubits. We show such an implementation of next-nearest-neighbour CZ gates in the first equality of Fig. 5(c) (this is adapted from Ref. [56]). This circuit also includes optional measurements, coloured in grey, which provide “flag information” for detecting Z errors on the ancilla qubit that may have propagated to the data qubits. Such flag measurements can assist in decoding [57]. In the second equality of Fig. 5(c), we reset the ancilla qubit in the $|0\rangle$ state instead of the $|+\rangle$ state, allowing for a shorter-depth circuit to implement the Z_1X_2 measurement. This comes at the cost of removing one flag measurement. To see that this circuit works as intended, we can commute the two-qubit gates past the X_2 measurement, resulting in a $Z_1Z_aX_2$ parity measurement. From the result of this measurement, together with the fact that Z_a value is set to be $+1$ at the ancilla reset, we can then infer the value of Z_1X_2 .

We emphasise that, even if the X measurements in the parity-check circuits above are implemented using Z measurements sandwiched by Hadamard gates, this does not degrade the bias, since a Z error occurring between those Hadamard gates is harmless. Specifically, a Z error happening before the measurement is immediately absorbed by the Z measurement without flipping its outcome, and a Z error after the measurement is a stabiliser of the state (up to a sign), so does nothing. Moreover, the

circuit will not change the noise bias significantly even if the Hadamard gates are noisy, since in many architectures, the single-qubit gate errors are not the predominant error source and are usually much smaller than the two-qubit gate errors [65–68].

In Supplementary Sec. II of the Supplementary Information [61], we show in more detail how the parity check circuits preserve the noise bias. We also show that the phase-bias preserving parity-check circuits presented in this paper have optimal circuit depth.

G. Memory experiment simulation

To study the performance of our proposed X^3Z^3 code, we perform quantum memory experiment simulations under two biased noise models. These noise models are the generalisations of the standard code-capacity and standard depolarising entangling measurement (SDEM3) noise models (see the Methods section for details of the noise models). We simulate the X^3Z^3 code along with three other codes: CSS, P^6 and XYZ^2 Floquet codes, with varying degrees of noise bias η (see the Methods section for details of our numerical simulations). The noise bias $\eta = p_Z/(p_X + p_Y)$ is defined as the ratio of the Z -error p_Z to other errors where the total physical error rate is $p = p_X + p_Y + p_Z$. As the noise asymmetry increases, the noise bias η increases from $\eta = 0.5$, which corresponds to fully depolarising noise ($p_X = p_Y = p_Z = p/3$), to $\eta = \infty$, corresponding to a pure Z -biased noise ($p_Z = p$, and $p_X = p_Y = 0$).

Thresholds

Figure 6 shows the thresholds of all codes for various levels of noise bias η calculated for (a) code-capacity and (b) SDEM3 noise models. Each threshold is obtained from the intersection of the logical failure probabilities p_L vs physical error rate p curves of different code distances $d_{\text{eff}} \equiv d/2 = 6, 8, 10$ and 12. Here, the effective distance d_{eff} is defined as the minimum number of faults under SDEM3 depolarising noise that produce a logical error. This is half of the distance d of the code-capacity noise model. For the methods and plots used to determine the thresholds, see the Methods section and Supplementary Figs. 5–8 of the Supplementary Information [61]. We also provide the threshold data in Ref. [69].

Since the code-capacity noise model, which considers only single-qubit noise, is a more benign model than the SDEM3 noise model, which also includes two-qubit and measurement errors, the code performance calculated under the code-capacity noise is better than that of the SDEM3 noise. For both noise models, as shown in Figs. 6(a) and 6(b), the performance of the X^3Z^3 code becomes increasingly better than those of all other tested codes as the noise bias increases. In particular, as the noise changes from fully depolarising to a pure dephasing type, the X^3Z^3 Floquet code's threshold increases from

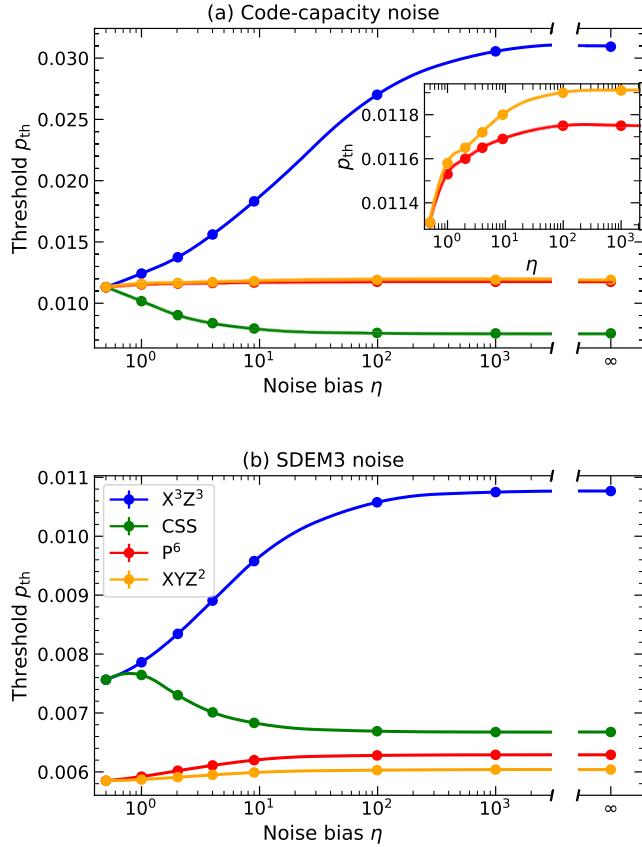


FIG. 6. Thresholds (p_{th}) of different Floquet codes as a function of noise bias η . Codes studied are X^3Z^3 (blue), CSS (green), P^6 (red) and XYZ^2 (orange). Results are computed for two different noise models: (a) code-capacity and (b) SDEM3. (a) Inset: Zoom-in threshold plots for the P^6 (red) and XYZ^2 (orange) codes. For better visualisation, we fit all curves with quadratic splines.

$\approx 1.13\%$ to $\approx 3.09\%$ for the code-capacity noise and from $\approx 0.76\%$ to $\approx 1.08\%$ for the SDEM3 noise model. The threshold therefore increases by a factor of 2.7 and 1.4 for the code capacity and SDEM3 noise models, respectively.

It is interesting to note that for the fully depolarising code-capacity noise, all codes have the same threshold. This is because at every measurement subround under this noise model, all the above codes have two types of errors at each fault location (occurring with a total probability $2p/3$) that give rise to edges in the detector graph and one type (occurring with probability $p/3$) that produces hyperedges. For the CSS and X^3Z^3 codes, these hyperedges result from Y Pauli errors. On the other hand, the hyperedge errors in the P^6 and XYZ^2 codes are those that anti-commute with the check measurements that occur just before and immediately after the errors [12, 22]. Therefore, the Pauli type of the hyperedge errors for the honeycomb codes varies between measurement subrounds. For example, Z errors create hyperedges only when they occur between the XX and

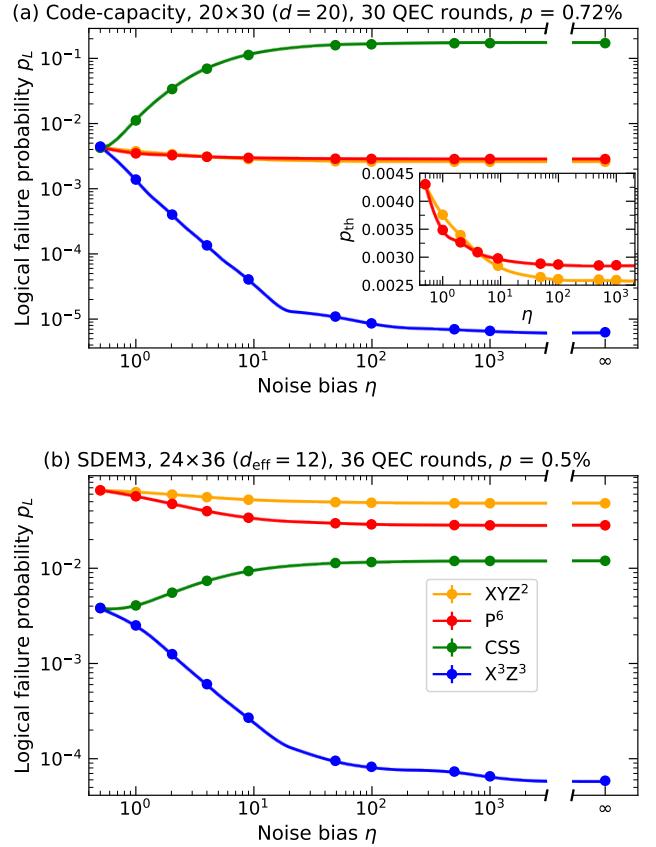


FIG. 7. Sub-threshold logical failure probability p_L of different Floquet codes as a function of noise bias η . Codes studied are X^3Z^3 (blue), CSS (green), P^6 (red) and XYZ^2 (orange). Results are computed for two different noise models: (a) code-capacity and (b) SDEM3. Inset: Zoom-in subthreshold performance plots of the P^6 (red) and XYZ^2 (orange) codes. Results are calculated using (a) lattice size 20×30 ($d = 20$) for 30 QEC rounds with $p = 0.72\%$ and (b) lattice size 24×36 ($d_{\text{eff}} = 12$) for 36 QEC rounds with $p = 0.5\%$. Each data point is averaged over $10^6 - 10^8$ Monte Carlo shots. For better visualisation, we fit all curves with quadratic splines. For each of the noise models, we choose the largest code distance and the smallest subthreshold physical error rate that can be simulated given our computational resources.

YY checks of the honeycomb codes. In summary, all the above codes have the same performance for fully depolarising code-capacity noise because their (weighted) detector hypergraphs are all equivalent.

As shown in Figs. 6(a) and 6(b), while the CSS Floquet code has the same threshold as the X^3Z^3 code when the noise is fully depolarising, its threshold decreases with increasing noise bias, where its value is 0.752% and 0.668% at infinite bias for the code-capacity and SDEM3 noise, respectively. This decrease is due to the fact that the CSS Floquet code has pure X and pure Z detectors where in the presence of biased noise, half of the detectors, i.e., those with the same type of the dominant error, will become less useful in detecting the biased errors.

Figures 6(a) and 6(b) show that the thresholds for the honeycomb codes have only minor improvements as the noise bias increases, where the thresholds for both honeycomb codes increase only by $\leq 6 \times 10^{-4}$. Specifically, the thresholds of the honeycomb codes at $\eta = \infty$ are only about 1.03 – 1.08 times larger than their thresholds at $\eta = 0.5$ (where their fully-depolarising noise thresholds are 1.13% and 0.585% for the code-capacity and SDEM3 models, respectively). We note that for the SDEM3 noise, the thresholds of both honeycomb codes are lower than those of the X^3Z^3 and CSS Floquet codes at all noise biases. This is partly explained by noting that the SDEM3 noise model contains measurement errors which give rise to hyperedges in the decoding hypergraphs of the honeycomb codes [12] but only graph-like edges for the CSS and X^3Z^3 Floquet codes [13]. As we explain below, these hyperedges degrade the MWPM decoder performance.

We find that there are differences, albeit modest ones, between the performance of the XYZ^2 and P^6 honeycomb codes under biased noise; these differences have not been pointed out before in the literature. While the XYZ^2 honeycomb code has a better performance than the P^6 honeycomb code for the code-capacity noise model, we find that its performance is inferior to that of the P^6 Floquet code for the SDEM3 noise model. This is surprising, considering the similarity of the former to the bias-tailored XYZ^2 static code [49]. However, owing to stabilisers being measured in two measurement subrounds, the XYZ^2 Floquet code no longer possesses all the symmetries of the static version making the dynamical code not able to inherit all the benefits of its static counterpart.

The performance difference between the XYZ^2 and P^6 honeycomb codes can be understood from the distribution of hyperedge-like syndromes (namely, those with four triggered detectors) in the detector hypergraphs under the biased SDEM3 noise model. These hyperedges generally degrade the code's performance when using a matching decoder since they must be decomposed into edges (see Ref. [12] for a detailed description of the decoding in a hyperbolic version of the honeycomb code). An infinitely phase-biased SDEM3 noise model that includes Pauli noise but no measurement errors can already form hyperedges in the honeycomb codes' detector hypergraphs. In particular, the P^6 code can have hyperedges resulting from Z errors occurring only after the red check measurement subrounds, while the hyperedges in XYZ^2 code can result from certain Z errors happening in any subround. While there is no difference in the overall number of hyperedges resulting from single-qubit Z errors in the two hypergraphs, the different arrangement of the hyperedges due to the single-qubit errors in these two codes gives rise to different kinds of syndromes for the two-qubit ZZ errors. Specifically, ZZ errors can lead to hyperedge syndromes in the XYZ^2 Floquet code but only edge-like syndromes for the P^6 code (see Supplementary Sec. IV

of the Supplementary Information [61] for details). As a result, in an infinitely phase-biased regime, where Pauli errors after MPP gates are evenly distributed between Z_1 , Z_2 and Z_1Z_2 , there are more hyperedge syndromes triggered in the XYZ^2 code than in the P^6 code. This is the reason underlying the difference in the thresholds, as shown in Fig. 6(b), of the two honeycomb codes under the SDEM3 noise. Since honeycomb codes are prone to hyperedge errors, better performance of the honeycomb codes might be expected using a correlated MWPM decoder such as the one used in Ref. [8].

H. Subthreshold performance

As a result of the shift of the thresholds with the noise bias, the subthreshold performance of the codes changes accordingly. As shown in Figs. 7(a) and 7(b), the subthreshold logical failure probability improves significantly for the X^3Z^3 code, while it becomes only slightly better for the honeycomb codes and deteriorates for the CSS code. In general, one expects the subthreshold performance of the codes to improve by increasing the code distance along the direction where the biased errors most likely form logical strings. For example, for the Z biased noise considered here, it would be by increasing the vertical code distance d_V . To this end, we study the subthreshold performance of elongated CSS and X^3Z^3 Floquet codes with $d_V > d_H$, where d_V and d_H are the vertical and horizontal code distances, respectively. The details are given in Supplementary Sec. V of the Supplementary Information [61].

Besides thresholds, another quantity of importance is the scaling of the code sub-threshold performance with the code distance. We plot the sub-threshold logical failure probability as a function of d or d_{eff} for varying levels of noise bias in Fig. 8. As shown, the logical failure probabilities for all codes decrease exponentially with the code distance, i.e., $p_L \propto \exp(-\gamma d)$ or $p_L \propto \exp(-\gamma d_{\text{eff}})$. Among all codes presented, the X^3Z^3 code has the largest logical error suppression rate γ . At higher noise bias, this error suppression rate becomes significantly larger for the X^3Z^3 code, moderately increases for the two honeycomb codes and decreases for the CSS Floquet code. The reason is that as the noise bias increases, a fixed subthreshold physical error rate moves relatively with respect to the shifting threshold, so that it becomes much further below the threshold for the X^3Z^3 Floquet code, moves moderately away from the threshold for the two honeycomb codes and becomes closer to the threshold for the CSS code [see Figs. 6(a) and 6(b)].

We also numerically simulate the performance of the X^3Z^3 Floquet code with a twisted periodic boundary condition. This twisted code has a pure Z -type logical operator with a length that scales quadratically with the code distance of the untwisted code. As a result, we find better thresholds and more favorable subthreshold per-

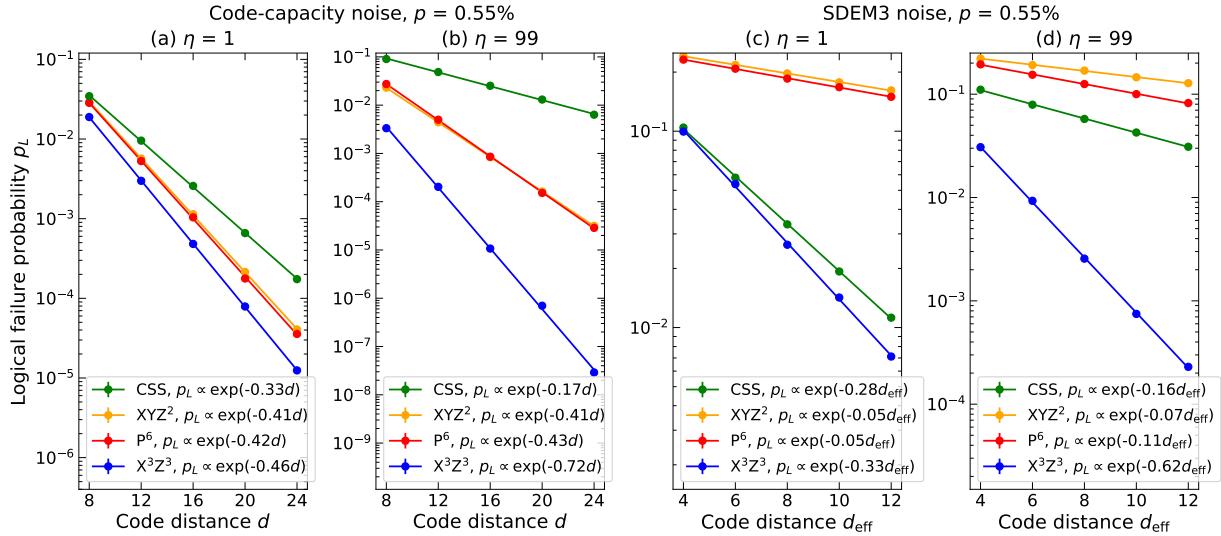


FIG. 8. Exponential suppression of the sub-threshold logical failure probabilities p_L with respect to code distance. We use code distances d or d_{eff} depending on the noise model and compare different Floquet codes: CSS (green), XYZ^2 (orange), P^6 (red), and X^3Z^3 (blue). Results are calculated for different noise models: (a,b) code-capacity and (c,d) SDEM3 noise models, with a subthreshold physical error rate $p = 0.55\%$, which is small enough, yet can still be simulated using our computational resources up to code distance $d = 24$ or $d_{\text{eff}} = 12$, for all the different codes and parameters corresponding to different curves in the plots. The same physical error rates are chosen for all curves such that the subthreshold performance under different noise bias strengths and noise models can be compared. Plots are computed using two different bias strengths; one representing noise near the depolarizing regime: (a,c) $\eta = 1$ and the other representing noise in the strongly dephasing regime: (b,d) $\eta = 99$. All curves can be fitted to an exponential decay function $f \propto \exp(-\gamma d)$ or $f \propto \exp(-\gamma d_{\text{eff}})$ where γ depends on the bias strength η and is an increasing function of $(p_{\text{th}} - p)$. Each data point is averaged over $10^5 - 10^9$ shots.

formance for the twisted code over the untwisted one. While these performance benefits hold for the *infinite-bias* code-capacity noise, they do not apply to other noise biases and models tested (See Supplementary Sec. VI of the Supplementary Information [61] for details).

While in this paper, we focus on Floquet codes with periodic boundary conditions, we expect similar qualitative performance of the Floquet codes under open boundary conditions. As shown in Ref. [9] for the honeycomb code with depolarizing noise, the boundary conditions (whether periodic or open) do not significantly impact the threshold and subthreshold performance of the code. This should be expected even more so for the CSS and X^3Z^3 Floquet codes, because the boundary condition does not affect the measurement schedule for the codes. We leave the simulations of the codes under open boundary conditions to future works.

The error suppression rate γ is related to the error suppression factor Λ introduced in Ref. [70] which is defined as the reduction factor in the logical failure probability as the code distance increases by 2. Mathematically, it is given by

$$\Lambda \equiv \frac{p_L(d)}{p_L(d+2)} = e^{2\gamma} \quad (4)$$

for the code-capacity noise and similarly for the SDEM3 noise model, but with d replaced by d_{eff} . For a noise model where the error is characterised by a single

	Code-capacity		SDEM3	
	$\eta = 1$	$\eta = 99$	$\eta = 1$	$\eta = 99$
CSS	1.94	1.39	1.74	1.37
XYZ^2	2.27	2.28	1.11	1.15
P^6	2.31	2.37	1.12	1.24
X^3Z^3	2.49	4.26	1.94	3.42

TABLE I. Error suppression factors Λ for different Floquet codes. These are calculated using two different noise models, code-capacity and SDEM3, with varying levels of noise bias η . Results are calculated with a physical error rate $p = 0.55\%$. The lambda factor is related to the error suppression rate γ shown in Fig. 8.

error rate p and threshold value p_{th} , when $p \ll p_{\text{th}}$, QEC suppresses the logical error exponentially as the code distance increases, i.e., $p_L \approx (p/p_{\text{th}})^{d/2}$. Since $\Lambda \equiv p_L(d)/p_L(d+2)$, we then have $\Lambda \approx p_{\text{th}}/p$. At the thresholds, we have $\Lambda(p_{\text{th}}) \approx 1$. The value of Λ increases as the physical error rate p moves further below the threshold p_{th} . For physical error rate below the threshold, $\Lambda > 1$ and larger Λ means greater error correction. Since the threshold depends on the noise bias η , Λ depends on both η and p . In Table I we list the values of Λ corresponding to the γ values shown in Fig. 8.

I. Performance optimality of X^3Z^3 Floquet code and no-go theorems

As seen above, although the performance of the X^3Z^3 Floquet code is significantly better compared to other Floquet codes, its threshold under infinitely phase-biased code-capacity noise does not reach the 50% threshold of its bias-tailored static version [46]. To explain this, we show that there are fewer symmetries in the decoding graphs of dynamical codes, which restricts the code performance. The key feature resulting in the high performance of bias-tailored static codes (i.e., their decoding can be understood as decoding a series of repetition codes [39, 40, 42, 46]) is not possible for dynamical codes built from two-qubit parity measurements, as we will show below.

We argue that due to the above limiting constraint, the symmetry of the X^3Z^3 Floquet code has likely rendered its performance close to optimal under a matching decoder, despite the fact that even for the most favorable case of infinitely phase-biased code-capacity noise, its decoding can only be reduced to at most a series of disjoint 2D (planar) graphs. We formalise this by proving two no-go theorems for the decoding graphs of dynamical codes. We begin with the following informal statement of our first theorem:

Theorem 1. *(Informal) A dynamical code (not necessarily a Floquet code) on the honeycomb lattice operated over a sufficiently large number of time steps cannot have a 1D decoding graph under an infinitely biased code-capacity noise model, so long as it obeys the following properties:*

- (a) *At each time step, two-body Pauli operators on edges of a given colour are measured. Measurements in consecutive time steps occur on edges of different colours and anti-commute.*
- (b) *Detectors are supported on plaquettes.*
- (c) *All non-trivial errors are detectable and produce syndromes of weight > 1 .*

In principle, a 1D graph [such as the one shown in Fig. 9(a)] could still result in better code performance than a general planar decoding graph. The fact that this type of graph is not possible suggests that planar graph decoding is already optimal for dynamical codes on the honeycomb lattice.

To generalise this result beyond codes defined on the honeycomb lattice, we provide the following theorem for general dynamical codes built from two-qubit parity measurements.

Theorem 2. *(Informal) A dynamical code (not necessarily a Floquet code) operated over a sufficiently long time scale cannot have a decoding graph, under an infinitely biased code-capacity noise model, equivalent to that of a collection of repetition codes [see Fig. 9(b)], so*

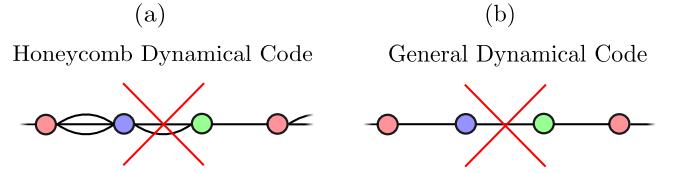


FIG. 9. **Illustrations of the no-go theorems.** We display examples of the decoding graphs (under an infinitely biased code-capacity noise model; edges correspond to Z errors and vertices to detectors) that are forbidden by (a) Theorem 1 for dynamical codes on the honeycomb lattice and (b) Theorem 2 for general dynamical codes.

long as: all measurements are two-qubit parity measurements, each qubit is in the support of one measurement in each time step, overlapping measurements in consecutive time steps anti-commute, and errors produce syndromes of weight > 1 .

This theorem shows that (given reasonable and typical assumptions) the decoding graphs of general dynamical codes cannot be decomposed into a collection of 1D repetition-code decoding graphs, which could otherwise allow them to achieve higher thresholds.

We provide the formal statements of the above two theorems and their proofs in the Methods section. We display the decoding graphs prohibited by the above two theorems in Fig. 9.

III. DISCUSSION

In this paper, we introduce the X^3Z^3 Floquet code, the first bias-tailored dynamical code based on two-qubit parity check measurements. We show that, despite having no constant stabilisers, the code possesses a persistent stabiliser-product symmetry under pure dephasing (or pure bit-flip) noise which allows for a simplified decoding. This results in a substantially improved threshold and sub-threshold performance under biased noise, when compared to other Floquet codes. We demonstrate the enhanced performance through our simulation results obtained from using a fast matching decoder and two error models: a simplistic code-capacity noise model and a noise model approaching realistic circuit-level errors. Besides the superior performance of the X^3Z^3 Floquet code, our results also show that there are differences, albeit modest ones, in the threshold and subthreshold performance between the XYZ^2 and P^6 honeycomb codes under biased noise.

To explain why the X^3Z^3 Floquet code does not reach the same high performance as the bias-tailored static codes, we prove that a dynamical code on the honeycomb lattice (obeying certain assumptions common to standard Floquet codes) cannot have a 1D decoding graph, the crucial requirement for the high performance of static codes. Despite this limitation, the bias-tailored X^3Z^3 Floquet

code has the advantage over its static counterpart in that it requires only lower-weight measurements. Specifically for devices without native two-qubit parity measurements, we devise phase-bias preserving parity check measurement circuits for any qubit architecture, which allow for high-performance implementation of the code. Our work therefore demonstrates that the X^3Z^3 Floquet code is a leading quantum error correction code especially for devices with limited connectivity such as the hexagonal and heavy-hexagonal architectures.

We now give several directions for future work. Since the decoding of any general Floquet codes with two-qubit parity measurements can be reduced, at most, to a 2D decoding problem, it would be interesting to investigate the performance improvements of the X^3Z^3 Floquet code using a more accurate decoder (such as a tensor network decoder) and to consider ways to analytically derive the best achievable thresholds for the X^3Z^3 Floquet code. Future studies may also investigate fault-tolerant logic in the X^3Z^3 code, via lattice surgery [7], aperiodic measurement sequences [21], twist braiding [71], or transversal gates [72]. Finally, it will be of interest to study the performance of memory experiments and fault-tolerant logical gate implementations in hardware where parity checks are implemented using bias-preserving circuits such as the circuits presented in this paper and compare them to the performance in hardware with native direct multi-Pauli product measurements.

IV. METHODS

A. Noise models

In this paper, we consider two noise models: code-capacity and entangling measurement (SDEM3) noise models. In the code-capacity noise model, we apply single-qubit Pauli noise on all the data qubits independently at every measurement subround. The single-qubit Pauli noise channel is given by

$$\mathcal{E}_{1q}(\rho) = (1-p)\rho + p_X \hat{X}\rho\hat{X} + p_Y \hat{Y}\rho\hat{Y} + p_Z \hat{Z}\rho\hat{Z}. \quad (5)$$

Here, $p \equiv p_X + p_Y + p_Z$ is the total error probability. As in the literature, we define the noise bias as $\eta = p_Z/(p_X + p_Y)$ and assume $p_X = p_Y$. Several values of η are worth listing:

1. $\eta = 0 \rightarrow p_Z = 0$, and $p_X = p_Y = p/2$,
2. $\eta = 0.5 \rightarrow p_X = p_Y = p_Z = p/3$,
3. $\eta = \infty \rightarrow p_Z = p$, and $p_X = p_Y = 0$.

We note that, since for Floquet codes one QEC round consists of several measurement subrounds, the single-qubit noise channel in the code-capacity model is applied several times in one QEC round instead of just once as is the case with static codes. The code-capacity noise model is often used as a preferred initial noise model

Noise model	Generalised SDEM3
Noisy gate set	$M_{PP}(p, \eta)$ Init(p, η) $M(p)$
Measurement ancilla	None
Measurement subrounds per one QEC round	6

TABLE II. Description of the generalised SDEM3 noise model. The definition of the noisy gates is given in Table III.

to study before going to a more involved model as its simplicity often offers insight into understanding the code performance.

On the other hand, the SDEM3 model is a more elaborate error model involving single-qubit noise channels after every single-qubit gate, measurement and reset, a two-qubit noise channel after every multi-Pauli product (MPP) parity measurement gate, and a classical flip after each measurement. As in Ref. [9], we assume that each of these error channels occurs with a total probability p . The SDEM3 noise model is therefore close to standard circuit-level noise and would be a more accurate description of a realistic noise channel, particularly in hardware with native two-qubit measurements, for example, Majorana [10, 62, 63] and photonic [17, 64] architectures.

To take into account biased noise, we generalise the SDEM3 model for depolarising noise used in Refs. [9, 15]. Here the single-qubit noise channel has the same form as given in Eq. (5) for the code-capacity noise model. On top of the single-qubit errors, this model also has a two-qubit noise channel applied after each of the MPP gates, which is given by

$$\mathcal{E}_{2q}(\rho) = (1-p)\rho + \sum_{O \in \{I, X, Y, Z\}^{\otimes 2} \setminus \{I \otimes I\}} p_O \hat{O}\rho\hat{O}. \quad (6)$$

To conform with how the bias is defined for the single-qubit noise channel, we also use η to characterize the bias of the two-qubit noise channel. Specifically, we define the bias η for the two-qubit noise channel such that

1. $\eta = 0 \rightarrow p_{ZZ} = p_{IZ} = p_{ZI} = 0$ and each of the other probabilities is $p/12$,
2. $\eta = 0.5 \rightarrow$ each of the Pauli errors occurs with $p/15$, and
3. $\eta = \infty \rightarrow p_{ZZ} = p_{IZ} = p_{ZI} = p/3$, and the other probabilities are 0.

Just as for the single-qubit noise, the above definition of η ensures that the two-qubit noise at the three special points, $\eta = 0$, $\eta = 0.5$ and $\eta = \infty$, are Z -error free, depolarising, and pure dephasing, respectively. Given that the two-qubit error probabilities at these η values must satisfy the conditions above, we define $\zeta \in [0, 1]$

Noisy gate set	Description
$M_{PP}(p, \eta)$	Measurement of Pauli Product PP on a pair of qubits: <ul style="list-style-type: none"> • independently register incorrect measurement result with probability p, • apply a two-qubit Z-biased noise channel with total error probability p and noise bias η after measurement, i.e., apply:<ul style="list-style-type: none"> – the trivial identity operator (II) with a probability $1 - p$, – ZI, IZ, and ZZ operators, each with probability $\zeta p/3$, – the remaining Pauli operators, each with probability $(1 - \zeta)p/12$, where $\zeta = \frac{3}{5} \left(\frac{\eta}{1+\eta} \right)^2 + \frac{2}{5} \left(\frac{\eta}{1+\eta} \right)$.
$\text{Init}(p, \eta)$	Qubit initialization in some Pauli basis, followed by applying a single-qubit Z -biased noise channel with total error probability p and noise bias η after the reset, i.e., apply: <ul style="list-style-type: none"> • the trivial identity operator (I) with a probability $1 - p$, • Pauli Z operator with probability $p\eta/(\eta + 1)$, • Pauli X and Y operators, each with probability $p/[2(\eta + 1)]$.
$M(p)$	Measure the qubit in some Pauli basis and register the incorrect measurement result with probability p .

TABLE III. Description of noisy gates in Table II.

and write the two-qubit Pauli error probabilities as

$$\begin{aligned} p_{ZZ} &= p_{ZI} = p_{IZ} = \zeta p/3, \\ p_O &= (1 - \zeta)p/12, \\ \text{for } O \in \{I, X, Y, Z\}^{\otimes 2} \setminus \{II, ZI, IZ, ZZ\}, \end{aligned} \quad (7)$$

with η and ζ related via:

$$\zeta = \frac{3}{5} \left(\frac{\eta}{1+\eta} \right)^2 + \frac{2}{5} \left(\frac{\eta}{1+\eta} \right). \quad (8)$$

Note that Eq. (7) is defined such that the total probability of all the Pauli errors $\sum_{O \in \{I, X, Y, Z\}^{\otimes 2} \setminus \{II, ZI, IZ, ZZ\}} p_O = p$ for all noise biases η . Apart from the single- and two-qubit Pauli noise channels, we also apply a classical flip of the measurement results with probability p after each of the single and two-qubit measurements. These measurement flips are uncorrelated with the single and two-qubit noise channels. We summarise the description of the SDEM3 noise model and its noisy gate set in Tables II and III, respectively.

B. Stabiliser circuits

We simulate the circuits and generate the error syndromes using Stim [73]; example Stim circuits are provided at Ref. [69]. We construct Stim circuits from various quantum operations (resets, measurement gates, etc.) with noise channels associated with each quantum operation (i.e., with error probabilities set by the noise models introduced above), detectors and logical observable updates. From these circuits, Stim can generate the detector error models which list the error mechanisms, the associated syndromes, and the

logical observables flipped by the errors. A detector error model is fed to the decoder which then predicts the most likely error based on a given syndrome. We refer the reader to Ref. [73] for more details on Stim.

C. MWPM decoder

To decode the error syndromes, we apply a minimum-weight perfect matching (MWPM) decoder [74], which is implemented using PyMatching [60]. The syndrome decoding is mapped by the MWPM decoder onto a graph problem, which is subsequently solved by utilizing Edmonds' algorithm [75, 76] for finding a perfect matching which has minimal weight, i.e., finding a minimum-total-weight set of edges, for which every vertex is incident to exactly one edge [77].

D. Details of numerical simulations

Our simulations are performed for four different Floquet codes using different values of physical error rates p and various strengths of noise bias η . Moreover, we consider two different noise models: code-capacity and SDEM3 noise models. For each code, the simulations are run with effective distances $d_{\text{eff}} \equiv d/2 = 4, 6, 8, 10, 12$, where the effective distance d_{eff} is defined as the minimum number of faults under SDEM3 depolarising noise that produce a logical error. This is half of the distance d of the code-capacity noise model. The calculated effective distance differs between noise models since in the SDEM3 noise model, two-qubit errors occurring after an

MPP parity gate count as a single fault, while under the code-capacity noise model they would count as two faults.

We choose lattices with periodic boundary conditions. For most simulations, we choose lattice sizes $L \times 3L/2$, where $L = d = 2d_{\text{eff}}$ is the lattice length which varies with respect to the code distance. We choose lattice sizes $L \times 3L/2$ such that the horizontal and vertical code distances are the same, as can be inferred from the minimum weight of the horizontal and vertical logical operators (see e.g., Fig. 1(c) for lattice length $L = 4$). We simulate the memory experiments for $3d_{\text{eff}}$ QEC rounds or $18d_{\text{eff}}$ measurement subrounds (1 QEC round consists of 6 measurement subrounds). For Supplementary Sec. V of the Supplementary Information [61] where we study elongated codes, we use lattice sizes of $d_H \times 3d_V/2$ where $d_V \geq d_H$. Here, d_H and d_V are the minimum weights of the logical operators in the horizontal and vertical directions, respectively. We simulate the horizontal and vertical logical memory experiments for $3d_H/2$ and $3d_V/2$ rounds, respectively.

We select a single logical qubit for each Floquet code (they all encode two logical qubits on a lattice with periodic boundary conditions) to test for the logical failure probability. Depending on the codes and parameter regimes, we run different numbers of Monte Carlo shots ranging from $10^5 - 10^{10}$ (larger number of shots for longer code distances and smaller physical error rates) for each of the horizontal and vertical logical observables, which give us the horizontal p_H and vertical p_V logical failure probabilities. We then report the combined logical error probability:

$$p_L = 1 - (1 - p_H)(1 - p_V), \quad (9)$$

which is an estimate of the probability that either a vertical or a horizontal logical error occurs. Equation (9) assumes that the horizontal and vertical logical errors occur independently. For the infinitely phase-biased code-capacity noise model, one of the logical errors which is of pure Z -type or pure X -type has zero logical error probability, because the pure biased noise cannot form the other logical operator which is of mixed-type and hence cannot flip the pure-type logical operator. As a result, for the infinitely phase-biased code-capacity noise, the maximum combined logical error probability is $p_L = 0.5$ [Eq. (9)]. This is in contrast to other noise biases and models where the maximum combined logical error probability is $p_L = 0.75$ since $p_V, p_H \leq 0.5$.

E. Determining the thresholds

To extract thresholds of the codes, we perform a finite-size collapse [78, 79] of the logical failure probability p_L data taken for various physical error rates p and code distances. This is done by fitting the data to the curve $A + Bx + Cx^2$ where $x = (p - p_{\text{th}})d^{1/\nu}$ or

$x = (p - p_{\text{th}})d_{\text{eff}}^{1/\nu}$ for the code-capacity and SDEM3 noise models, respectively. Here, ν is the critical exponent, p_{th} is the Pauli threshold, A, B, C are the fit parameters, d and d_{eff} are code distances under the code-capacity and SDEM3 noise models, respectively. This data is presented in Supplementary Figs. 5-8 of the Supplementary Information [61].

F. No-go theorems for 1D decoding graphs of dynamical codes

Here, we provide formal statements and proofs for the two no-go theorems presented in the Results section. To this end, we begin by providing several definitions.

Definition 1. Let V be a set of detectors for a dynamical code. For every qubit and every time step (q, t) , i.e., every fault location in the code, define a hyperedge $e_{(q,t)} = (v_1, v_2, \dots)$, where the $v_i \in V$ are detectors that return -1 outcomes if a single Z error occurs at the fault location (q, t) . The **Z-detector hypergraph** (ZDH) is defined as $G = (V, E)$, where $E = \bigcup_{q,t} e_{(q,t)}$ is the set of all hyperedges. A Z-detector graph (ZDG) is a Z-detector hypergraph in which all hyperedges are edges (their size is 2).

Definition 2. A **1D-decodable** Z-detector graph is one in which all vertices have neighbourhoods of size no greater than 2.

Finally, we define dynamical codes on the honeycomb lattice in the following way:

Definition 3. A honeycomb dynamical code (HDC), with “duration” T and constant “initial(final)-time boundary offsets” t_i (t_f), is a finite-depth measurement circuit acting on qubits of the honeycomb lattice (without boundary). The sets of measurements in the circuit, \mathcal{M}_t ($t \in \{1, \dots, T\}$), are composed of two-body Pauli measurements along coloured edges (either r , g , or b , for each time step) of the lattice, performed sequentially on a state stabilised by some group \mathcal{S} . The HDC obeys the following properties:

- (a) Overlapping measurements in consecutive time steps anti-commute and are supported on different edges.
- (b) Detectors (for time steps $t > t_i$) are associated with plaquettes in the lattice (they have support only on qubits around their associated plaquette).
- (c) If a single-qubit or two-qubit error occurring in time steps $t_i < t < T - t_f$ anti-commutes with future measurements, it is detectable, unless the two-qubit error is the same as an edge operator just measured or to be measured in the next time step.
- (d) All single-qubit errors in time steps $t_i < t < T - t_f$ have syndromes of weight > 1 .

Let us first comment on our definition of an HDC. The chosen properties are very natural. The anti-commutation of consecutive measurement subroutines ensures “local reversibility” of the code, so as to preserve the quantum information and its locality [80] (although note that we do not require the code to have logical qubits in our definition). The requirement that consecutive sets of measurements are supported on different edges is not very restrictive: if two (anti-commuting) measurements act on the same edge in consecutive time steps, we can replace the second with a Clifford gate and commute that to the end of the circuit. This merely changes the bases of subsequent measurements without changing their (anti-)commutation. Since the Clifford gates are unimportant, we can ignore them and hence we end up with a shorter duration T . We restrict our attention to errors occurring in the temporal “bulk” of the code, $t_i < t < T - t_f$. This avoids complications due to errors occurring close to the initial and final-time boundaries where detectors here may be formed in different ways than the bulk detectors. Specifically, the detectors associated with errors at the initial and final time boundaries are obtained by comparing the edge or plaquette measurement values with the initial state and final read-out of the physical qubits, respectively. The exclusion of these final time boundary detectors is also required here since we do not include final read-out measurements in our definition of an HDC. In summary, we consider errors occurring in all “detection cells” [13] that are completed before time step T and begin at time step $t \geq 1$, where a detection cell consists of all the spacetime points at which a detector can identify errors.

We will show that an HDC cannot even have a 1D-decodable ZDG, which implies that its decoding graph cannot be equivalent to a collection of the much simpler repetition code’s graphs. Note that a repetition code graph is not only 1D-decodable but also has a maximum degree of 2, which means there are no double edges between neighbouring vertices. Here, however, we allow for these double edges in our definition of 1D-decodability, and show that this more general property is also impossible. Such double edges naturally arise in Floquet codes, where they correspond to two-qubit undetectable errors (i.e., edge operators just measured or about to be measured).

We begin by first proving the following lemma which is going to be used in the proof of Theorem 1.

Lemma 1. *In a honeycomb dynamical code, at each time step $t > 1$, no detectors can be formed on neighbouring plaquettes.*

Proof. Without loss of generality, suppose neighbouring plaquettes F_1 and F_2 are coloured red and blue, respectively. Suppose, for the sake of contradiction, that they both host detectors at a time step $t > 1$. Let $\mathcal{D}_1, \mathcal{D}_2 \subset \mathcal{M}_t$ be the sets of check measurements from \mathcal{M}_t whose products around the plaquettes F_1 and F_2 , respectively, form the detectors at the corresponding pla-

lettes at time t . That is, the measurements in \mathcal{D}_j act only on qubits around F_j . Since at every time step, only edges of one particular colour are being measured and green edges are shared by blue and red plaquettes, the measurement set \mathcal{M}_t at time t can contain only measurements on green edges. Otherwise, either \mathcal{D}_1 or \mathcal{D}_2 would have to be empty or they would include measurements on qubits not adjacent to F_1 or F_2 , respectively, which is not allowed by the definition of \mathcal{D}_1 and \mathcal{D}_2 .

To form detectors at both plaquettes, we require that, for all $M \in \mathcal{M}_{t-1}$, $[M, \prod_{D \in \mathcal{D}_j} D] = 0$, for $j = 1, 2$. Otherwise the plaquette operator outcomes would be deterministic. However, from property (a) of Definition 3 of an HDC, for each $D \in \mathcal{D}_1$, there is a measurement in \mathcal{M}_{t-1} that anti-commutes with it. Therefore, $\prod_{D \in \mathcal{D}_j} D$ must have even overlap with all edge measurements in \mathcal{M}_{t-1} , for $j = 1, 2$. This condition can only be satisfied for both $j = 1, 2$ if all $M \in \mathcal{M}_{t-1}$ are also on green edges (if they are supported on red edges, there will be one measurement with odd overlap with each $D \in \mathcal{D}_1$, and similarly for $D \in \mathcal{D}_2$ if they are supported on blue edges). But now both \mathcal{M}_{t-1} and \mathcal{M}_t are made up of measurements on green edges, contradicting property (a) of Definition 3. \square

Using the properties of the HDC as defined above, we now present the first main theorem of our paper:

Theorem 1. *A honeycomb dynamical code with duration more than $3t_f + t_i + 3$ cannot have a 1D-decodable Z-detector graph, where t_i and t_f are the initial and final time-boundary offsets.*

Since the proof of the theorem is quite long, here we begin by providing the sketch of the proof. We first show that the Z-detector graph of an HDC contains many more edges than vertices. This means the graph must contain many cycles. We will then find a *subgraph* of the ZDG in which no two vertices are connected by more than one edge, where for such a subgraph, the size of the neighbourhood of each vertex is equal to the number of edges incident to it. In particular, we will show that there exists at least one such subgraph that *still* possesses many more edges than vertices, meaning it is not possible for all vertices in the subgraph to have neighbourhoods of size ≤ 2 – if that were the case, then there would be at least as many vertices as edges. This in turn implies that the full ZDG must contain some vertices with neighbourhoods of size > 2 .

The full proof of Theorem 1 is as follows.

Proof. We take an error occurring in time step t to mean that it occurs immediately *after* the measurements in \mathcal{M}_t . Suppose there are T time steps and n qubits in the HDC. Let us consider part of the ZDG that contains vertices corresponding to all detectors formed in time steps $t > t_i$, but only those edges corresponding to *detectable* errors, i.e., those occurring in time steps $t_i < t < T - t_f$. We will call this subgraph $\overline{\text{ZDG}}$. The number of vertices

in this subgraph, corresponding to the number of detectors formed, is at most $n(T-t_i)/6$, since at each time step no detectors can be formed on neighbouring plaquettes (see Lemma 1), and there are $n/6$ such non-neighbouring plaquettes, which correspond to all plaquettes of a single colour, in the honeycomb lattice without boundary. Meanwhile, Z errors generate n edges between each time step, resulting in a total of $n(T-1-t_i-t_f)$ edges in the ZDG.

Let us now create a bipartition of the qubits into sets C and D such that no qubits in C are adjacent to one another, and similarly for qubits in D . This is possible since the honeycomb lattice is bipartite. Let us consider the subgraph of $\overline{\text{ZDG}}$, denoted by $\overline{\text{ZDG}}_C$, that contains only edges corresponding to Z errors on qubits in C . The subgraph $\overline{\text{ZDG}}_C$ therefore contains $n(T-1-t_i-t_f)/2$ edges. Suppose two vertices are connected in this graph by more than one edge. In such a scenario, we can choose any two from the collection of edges between these vertices. These two edges correspond to a weight-2 error, i.e., an error with support only on two qubits, in C , which is undetectable. This error is undetectable because the combination of the two errors, where each flips the values of the same two detectors, triggers no detector.

Consider one such undetectable weight-2 error, $Z_{i,t}Z_{j,t'}$, where $Z_{i,t}$ denotes an error that acts on qubit i at time step t and similarly for $Z_{j,t'}$. We will show that for $Z_{i,t}Z_{j,t'}$ not to be detectable, the two errors must act on the same qubit ($i = j$) but at adjacent time steps ($t = t' \pm 1$). To this end, we begin by ruling out other possibilities. First, we show that if the qubits $i, j \in C$ are different then the error $Z_{i,t}Z_{j,t'}$ will anti-commute with some future measurements. This, along with the fact that $Z_{i,t}Z_{j,t}$ is not an operator just measured or about to be measured (since i and j are not connected by an edge), means that the error is detectable by property (c) of Definition 3 of an HDC. We begin by analysing the case where the two errors occur at the same time step ($t = t'$). Suppose that this error commutes with all $M \in \mathcal{M}_{t+1}$. Since every measurement in \mathcal{M}_{t+1} must have support only on one qubit in C , then the two check measurements in \mathcal{M}_{t+1} , one with support on qubit i and the other on qubit j , must have Pauli Z operator supports on those respective qubits. Now consider the measurement $M' \in \mathcal{M}_{t+2}$ with support on qubit i . By property (a) of Definition 3, one measurement from the set \mathcal{M}_{t+2} that has a support on qubit i cannot act as a Pauli Z operator on that qubit, since it has to anti-commute with the previous edge measurement from \mathcal{M}_{t+1} that also has a Z support on the same qubit. Therefore, M' will anti-commute with $Z_{i,t}Z_{j,t}$ which, according to property (c) of Definition 3, means that the error $Z_{i,t}Z_{j,t}$ is detectable. Similarly, if the errors occur on different qubits but at different time steps, they must also be detectable. To show this, we can argue using the reasons mentioned above that $Z_{i,t}$ must anti-commute with a future measurement in set \mathcal{M}_{t+1} or \mathcal{M}_{t+2} . Since qubits i and j are not connected by an edge in the honeycomb lattice, the same

measurement, however, cannot anti-commute with $Z_{j,t'}$.

Using the above reasonings, we can infer that the only possibility for the weight-2 error to be undetectable is for the two errors to occur on the same qubit, but at different time steps, i.e., $Z_{i,t}Z_{i,t'}$ (note that if the two errors on qubit i occur at the same time step then they cancel each other out, which is equivalent to there being no error at all). Suppose, without loss of generality, that $t < t'$. For the weight-2 error to be undetectable, the measurements in between them must commute with Z_i . Since consecutive measurements with support on the same qubit anti-commute, the two errors acting on qubit i must be separated only by one measurement time step where the in-between measurement commutes with the two errors.

The above analysis implies that vertices in $\overline{\text{ZDG}}_C$ can be connected by only at most two edges. All such edges would have to correspond to errors on the same qubit and if there were more than two, we could find a pair of edges that correspond to an undetectable error $Z_{i,t}Z_{i,t'}$ with $t' > t + 1$, which we have shown above to be impossible. Therefore, at most half of the edges in $\overline{\text{ZDG}}_C$ can be removed without changing which vertices are connected. This means that there is a spanning subgraph of $\overline{\text{ZDG}}_C$ with at least $\lceil n(T-1-t_i-t_f)/4 \rceil$ edges with no vertices in the subgraph connected by more than one edge.

All things considered, this implies that the full ZDG has a subgraph with $\lceil n(T-1-t_i-t_f)/4 \rceil$ edges and a number of vertices $\leq n(T-t_i)/6$, such that no two vertices are connected by more than one edge. For $T > 3t_f + t_i + 3$, there are more edges than vertices in this subgraph, which means that it cannot have all vertices with neighbourhoods of size ≤ 2 . Therefore, some vertices in the full ZDG also must have neighbourhoods with size greater than 2. \square

The above theorem implies that no matter how an HDC is bias-tailored, it likely can never achieve as high a threshold as the repetition code or the bias-tailored stabiliser codes. Indeed, for large T ($T \gg t_i, t_f$), there will be at least approximately 6 times as many edges as vertices in the ZDG, and there is a subgraph with $3/2$ as many edges as vertices without any pair of vertices sharing more than one edge. This means the graph must have many cycles with length at least three edges.

It is clear that the codes examined in this paper obey the properties of an HDC. In particular, any Floquet codes related to the CSS Floquet code by Hadamard gates applied to a subset of the qubits (as is the case for the X^3Z^3 Floquet code) do not have 1D-decodable ZDGs. The reason we have focused on dynamical codes that have ZDGs, e.g., Hadamard-deformed CSS codes, instead of codes with hyperedges in their Z -detector hypergraphs, is because their decoding graphs are immediate generalisations of that of the CSS code. The fact that 1D-decodable ZDGs are not possible for an HDC suggests that there are not enough independent symmetries to reduce its decoding graph to the extent possible for sta-

biliser codes. It is interesting to note that even with hyperedges in the Z -detector hypergraph, a stabiliser code can still be bias-tailored [39] such that it possesses large enough symmetries to render its thresholds close to the hashing bound [81, 82]. While such high-thresholds are possible for static codes, we do not believe this would be the case for HDCs.

Finally, we consider more general codes defined on an arbitrary graph G .

Definition 4. *A general dynamical code (GDC) is a finite-depth measurement circuit acting on qubits associated with vertices of a graph G . It has a duration T and initial (final) time boundary offsets t_i (t_f), as defined above. The sets of measurements \mathcal{M}_t are two-body Pauli measurements associated with edges of G (each qubit is in the support of one measurement in each time step), performed sequentially on a state stabilised by some group \mathcal{S} . A GDC obeys properties (a) and (d) of an HDC.*

Having defined GDCs, we now provide the second main theorem of our paper:

Theorem 2. *A GDC with a duration at least $2t_f + t_i + 2$ (i.e., a constant), where t_i and t_f are the initial and final time-boundary offsets, respectively, cannot have a Z -detector graph equivalent to a collection of disjoint repetition codes.*

Proof. Since there are $n/2$ measurements in a given time step, there can be at most $n/2$ independent detectors formed from such measurements. However, there will in fact be far fewer for a GDC, because each individual measurement anti-commutes with a measurement from the previous time step resulting in a random value for each edge measurement. Therefore, a detector with a deterministic outcome must be formed from at least a product of two non-deterministic measurements within a time step. Suppose the GDC has a Z -detector graph (edges in the ZDG correspond to particular Z errors). Again, we consider the subgraph corresponding to Z errors occurring in time steps $t_i < t < T - t_f$, where T is the GDC's duration, and detectors formed in time steps $t > t_i$. There are $< n(T - t_i)/2$ vertices and $n(T - 1 - t_i - t_f)$ edges in this subgraph. Since the number of vertices of a collection of repetition code decoding graphs cannot be less than the number of edges, this means that the ZDG of a GDC cannot be a collection of repetition code graphs for $T \geq 2t_f + t_i + 2$. \square

Owing to this theorem, we do not expect to see thresholds in dynamical codes as high as 50%. The reason that static codes perform better than dynamical codes under biased noise is that they obey two separate symmetries under an infinitely phase-biased code-capacity noise model. There is a spatial symmetry

that forces syndromes to appear in pairs along 1D strips of the lattice, but there is also a temporal symmetry: syndromes must appear in pairs within a given time slice. These two symmetries simplify the detector graph into a collection of disjoint repetition code decoding graphs, leading to a high performance (up to 50% threshold) of the static codes under biased noise. On the other hand, even for the case where measurement errors are not considered, GDCs necessarily break the temporal symmetry (for HDCs, this is a consequence of Lemma 1), leading to edges between vertices in different time slices of the detector graph (see Fig. 3). We thus expect that such codes can only possess a single symmetry under an infinitely phase-biased code-capacity noise model, as the X^3Z^3 Floquet code does.

ACKNOWLEDGMENTS

We would like to thank our colleagues at Riverlane, especially Alexandra Moylett, Ophelia Crawford, and György Gehér for helpful discussions. We are also grateful to Ophelia Crawford and Hari Krovi for their comments on the manuscript. CM would like to thank Benjamin Brown for pointing out a useful circuit for performing next-nearest-neighbour entangling gates. CM acknowledges support from the Intelligence Advanced Research Projects Activity (IARPA), under the Entangled Logical Qubits program through Cooperative Agreement Number W911NF-23-2-0223.

Author contributions

FS developed the X^3Z^3 Floquet code and biased noise models. FS also performed the numerical simulations for all Floquet codes investigated. CM characterised the symmetry and decoding features of the codes tested. In addition, CM developed the bias-tailored circuits and the proofs of the no-go theorems. Both authors contributed to the writing of the manuscript.

Data availability

The threshold data is available in <https://doi.org/10.5281/zenodo.14258878> (Ref. [69]). Other numerical data generated in this work is available from the authors upon reasonable request. The video talk on this manuscript is available in <https://www.youtube.com/watch?v=nqQT-5IRC9w>.

Code availability

Circuits used in the simulations are available in <https://doi.org/10.5281/zenodo.14258878> (Ref. [69]).

Competing interests

The authors declare no competing interests.

-
- [1] P. W. Shor, Scheme for reducing decoherence in quantum computer memory, *Phys. Rev. A* **52**, R2493 (1995).
- [2] P. W. Shor, Fault-tolerant quantum computation, in *Proceedings of 37th conference on foundations of computer science* (IEEE, 1996) p. 56.
- [3] A. R. Calderbank and P. W. Shor, Good quantum error-correcting codes exist, *Phys. Rev. A* **54**, 1098 (1996).
- [4] S. J. Devitt, W. J. Munro, and K. Nemoto, Quantum error correction for beginners, *Reports on Progress in Physics* **76**, 076001 (2013).
- [5] D. Gottesman, Opportunities and challenges in fault-tolerant quantum computation, in *The Physics of Quantum Information, ed. David Gross, Alexander Sevin, and Peter Zoller, Proceedings of the 28th Solvay Conference on Physics* (World Scientific, 2023) pp. 287–309.
- [6] M. B. Hastings and J. Haah, Dynamically generated logical qubits, *Quantum* **5**, 564 (2021).
- [7] J. Haah and M. B. Hastings, Boundaries for the honeycomb code, *Quantum* **6**, 693 (2022).
- [8] C. Gidney, M. Newman, A. Fowler, and M. Broughton, A fault-tolerant honeycomb memory, *Quantum* **5**, 605 (2021).
- [9] C. Gidney, M. Newman, and M. McEwen, Benchmarking the planar honeycomb code, *Quantum* **6**, 813 (2022).
- [10] A. Paetznick, C. Knapp, N. Delfosse, B. Bauer, J. Haah, M. B. Hastings, and M. P. da Silva, Performance of planar Floquet codes with Majorana-based qubits, *PRX Quantum* **4**, 010310 (2023).
- [11] O. Higgott and N. P. Breuckmann, Constructions and performance of hyperbolic and semi-hyperbolic Floquet codes, *PRX Quantum* **5**, 040327 (2024).
- [12] A. Fahimniya, H. Dehghani, K. Bharti, S. Mathew, A. J. Kollár, A. V. Gorshkov, and M. J. Gullans, Fault-tolerant hyperbolic Floquet quantum error correcting codes, *arXiv:2309.10033* (2023).
- [13] M. S. Kesselring, J. C. Magdalena de la Fuente, F. Thomassen, J. Eisert, S. D. Bartlett, and B. J. Brown, Anyon condensation and the color code, *PRX Quantum* **5**, 010342 (2024).
- [14] S. Gu, A. Retzker, and A. Kubica, Fault-tolerant quantum architectures based on erasure qubits, *Phys. Rev. Res.* **7**, 013249 (2025).
- [15] C. McLauchlan, G. P. Gehér, and A. E. Moylett, Accommodating fabrication defects on Floquet codes with minimal hardware requirements, *Quantum* **8**, 1562 (2024).
- [16] J. C. Magdalena de la Fuente, J. Old, A. Townsend-Teague, M. Rispler, J. Eisert, and M. Müller, XYZ ruby code: Making a case for a three-colored graphical calculus for quantum error correction in spacetime, *PRX Quantum* **6**, 010360 (2025).
- [17] P. Hilaire, T. Dessertaine, B. Bourdoncle, A. Denys, G. de Gliniasty, G. Valentí-Rojas, and S. Mansfield, Enhanced fault-tolerance in photonic quantum computing: Floquet code outperforms surface code in tailored architecture, *arXiv:2410.07065* (2024).
- [18] M. Davydova, N. Tantivasadakarn, and S. Balasubramanian, Floquet codes without parent subsystem codes, *PRX Quantum* **4**, 020341 (2023).
- [19] A. Dua, N. Tantivasadakarn, J. Sullivan, and T. D. El-lison, Engineering 3D Floquet codes by rewinding, *PRX Quantum* **5**, 020305 (2024).
- [20] M. S. Alam and E. Rieffel, Dynamical logical qubits in the Bacon-Shor code, *arXiv:2403.03291* (2024).
- [21] M. Davydova, N. Tantivasadakarn, S. Balasubramanian, and D. Aasen, Quantum computation from dynamic automorphism codes, *Quantum* **8**, 1448 (2024).
- [22] X. Fu and D. Gottesman, Error correction in dynamical codes, *arXiv:2403.04163* (2024).
- [23] C. T. Aitchison and B. Béri, Competing automorphisms and disordered floquet codes, *Phys. Rev. B* **111**, 235112 (2025).
- [24] A. Y. Kitaev, Fault-tolerant quantum computation by anyons, *Annals of physics* **303**, 2 (2003).
- [25] 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).
- [26] D. Aasen, J. Haah, P. Bonderson, Z. Wang, and M. Hastings, Fault-tolerant Hastings-Haah codes in the presence of dead qubits, *arXiv:2307.03715* (2023).
- [27] X. Sun, L. Li, Z. Wu, Z. Guo, P. Huang, W. Huang, Q. Li, Y. Liang, Y. Liu, D. Sun, *et al.*, Universal logical operations with a dynamical qubit in Floquet code, *arXiv:2503.03867* (2025).
- [28] B. Hetényi and J. R. Wootton, Tailoring quantum error correction to spin qubits, *Phys. Rev. A* **109**, 032433 (2024).
- [29] R. Lescanne, M. Villiers, T. Peronnin, A. Sarlette, M. Delbecq, B. Huard, T. Kontos, M. Mirrahimi, and Z. Leghtas, Exponential suppression of bit-flips in a qubit encoded in an oscillator, *Nature Physics* **16**, 509–513 (2020).
- [30] A. Grimm, N. E. Frattini, S. Puri, S. O. Mundhada, S. Touzard, M. Mirrahimi, S. M. Girvin, S. Shankar, and M. H. Devoret, Stabilization and operation of a Kerr-cat qubit, *Nature* **584**, 205 (2020).
- [31] G. de Gliniasty, P. Hilaire, P.-E. Emeriau, S. C. Wein, A. Salavrakos, and S. Mansfield, A spin-optical quantum computing architecture, *Quantum* **8**, 1423 (2024).
- [32] S. J. Evered, D. Bluvstein, M. Kalinowski, S. Ebadi, T. Manovitz, H. Zhou, S. H. Li, A. A. Geim, T. T. Wang, N. Maskara, *et al.*, High-fidelity parallel entangling gates on a neutral-atom quantum computer, *Nature* **622**, 268 (2023).
- [33] T. Watson, S. Philips, E. Kawakami, D. Ward, P. Scarlino, M. Veldhorst, D. Savage, M. Lagally, M. Friesen, S. Coppersmith, *et al.*, A programmable two-qubit quantum processor in silicon, *Nature* **555**, 633 (2018).
- [34] T. Tanttu, W. H. Lim, J. Y. Huang, N. Dumoulin Stuyck, W. Gilbert, R. Y. Su, M. Feng, J. D. Cifuentes, A. E. Seedhouse, S. K. Seritan, *et al.*, Assessment of the errors of high-fidelity two-qubit gates in silicon quantum dots, *Nature Physics* **20**, 1804 (2024).
- [35] D. Aasen, M. Hell, R. V. Mishmash, A. Higginbotham, J. Danon, M. Leijnse, T. S. Jespersen, J. A. Folk, C. M. Marcus, K. Flensberg, and J. Alicea, Milestones toward Majorana-based quantum computing, *Phys. Rev. X* **6**, 031016 (2016).
- [36] C. McLauchlan and B. Béri, A new twist on the Majorana surface code: Bosonic and fermionic defects for fault-tolerant quantum computation, *Quantum* **8**, 1400 (2024).
- [37] D. Aasen, M. Aghaei, Z. Alam, M. Andrzejczuk, A. Antipov, M. Astafiev, L. Avilovas, A. Barzegar,

- B. Bauer, J. Becker, *et al.*, Roadmap to fault tolerant quantum computation using topological qubit arrays, [arXiv:2502.12252](https://arxiv.org/abs/2502.12252) (2025).
- [38] O. Higgott, T. C. Bohdanowicz, A. Kubica, S. T. Flammia, and E. T. Campbell, Improved decoding of circuit noise and fragile boundaries of tailored surface codes, *Phys. Rev. X* **13**, 031007 (2023).
- [39] D. K. Tuckett, S. D. Bartlett, and S. T. Flammia, Ultra-high error threshold for surface codes with biased noise, *Phys. Rev. Lett.* **120**, 050505 (2018).
- [40] 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, *Phys. Rev. X* **9**, 041031 (2019).
- [41] 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, *Phys. Rev. Lett.* **124**, 130501 (2020).
- [42] J. P. Bonilla Ataides, D. K. Tuckett, S. D. Bartlett, S. T. Flammia, and B. J. Brown, The XZZX surface code, *Nature Communications* **12**, 2172 (2021).
- [43] Q. Xu, N. Mannucci, A. Seif, A. Kubica, S. T. Flammia, and L. Jiang, Tailored XZZX codes for biased noise, *Phys. Rev. Res.* **5**, 013035 (2023).
- [44] J. F. San Miguel, D. J. Williamson, and B. J. Brown, A cellular automaton decoder for a noise-bias tailored color code, *Quantum* **7**, 940 (2023).
- [45] A. Dua, A. Kubica, L. Jiang, S. T. Flammia, and M. J. Gullans, Clifford-deformed surface codes, *PRX Quantum* **5**, 010347 (2024).
- [46] K. Tiurev, A. Pesah, P.-J. H. S. Derkx, J. Roffe, J. Eisert, M. S. Kesselring, and J.-M. Reiner, Domain wall color code, *Phys. Rev. Lett.* **133**, 110601 (2024).
- [47] D. Forlivesi, L. Valentini, and M. Chiani, Quantum codes for asymmetric channels: ZZZY surface codes, *IEEE Communications Letters* **28**, 2233 (2024).
- [48] P. K. Sarvepalli, A. Klappenecker, and M. Rötteler, Asymmetric quantum codes: constructions, bounds and performance, *Proceedings of the Royal Society A: Mathematical, Physical and Engineering Sciences* **465**, 1645 (2009).
- [49] B. Srivastava, A. Frisk Kockum, and M. Granath, The XYZ^2 hexagonal stabilizer code, *Quantum* **6**, 698 (2022).
- [50] J. Roffe, L. Z. Cohen, A. O. Quintavalle, D. Chandra, and E. T. Campbell, Bias-tailored quantum LDPC codes, *Quantum* **7**, 1005 (2023).
- [51] J. Claes, J. E. Bourassa, and S. Puri, Tailored cluster states with high threshold under biased noise, *npj Quantum Information* **9**, 9 (2023).
- [52] E. Huang, A. Pesah, C. T. Chubb, M. Vasmer, and A. Dua, Tailoring three-dimensional topological codes for biased noise, *PRX Quantum* **4**, 030338 (2023).
- [53] K. Sahay, J. Jin, J. Claes, J. D. Thompson, and S. Puri, High-threshold codes for neutral-atom qubits with biased erasure errors, *Phys. Rev. X* **13**, 041013 (2023).
- [54] K. Sahay, J. Claes, and S. Puri, Tailoring fusion-based error correction for high thresholds to biased fusion failures, *Phys. Rev. Lett.* **131**, 120604 (2023).
- [55] D. Forlivesi, L. Valentini, and M. Chiani, Logical error rates of XZZX and rotated quantum surface codes, *IEEE Journal on Selected Areas in Communications* **42**, 1808 (2024).
- [56] B. Hetényi and J. R. Wootton, Creating entangled logical qubits in the heavy-hex lattice with topological codes, *PRX Quantum* **5**, 040334 (2024).
- [57] C. Chamberland, G. Zhu, T. J. Yoder, J. B. Hertzberg, and A. W. Cross, Topological and subsystem codes on low-degree graphs with flag qubits, *Phys. Rev. X* **10**, 011022 (2020).
- [58] S. Puri, L. St-Jean, J. A. Gross, A. Grimm, N. E. Frattini, P. S. Iyer, A. Krishna, S. Touzard, L. Jiang, A. Blais, *et al.*, Bias-preserving gates with stabilized cat qubits, *Science advances* **6**, eaay5901 (2020).
- [59] E. Dennis, A. Kitaev, A. Landahl, and J. Preskill, Topological quantum memory, *Journal of Mathematical Physics* **43**, 4452 (2002).
- [60] O. Higgott and C. Gidney, Sparse blossom: correcting a million errors per core second with minimum-weight matching, *Quantum* **9**, 1600 (2025).
- [61] See Supplementary Information for details on the honeycomb and CSS Floquet codes, parity-check circuits, plots for determining thresholds, hyperedge errors, simulation results for Floquet codes with elongated dimension and a twisted boundary condition.
- [62] T. Karzig, C. Knapp, R. M. Lutchyn, P. Bonderson, M. B. Hastings, C. Nayak, J. Alicea, K. Flensberg, S. Plugge, Y. Oreg, C. M. Marcus, and M. H. Freedman, Scalable designs for quasiparticle-poisoning-protected topological quantum computation with Majorana zero modes, *Phys. Rev. B* **95**, 235305 (2017).
- [63] C. K. McLauchlan and B. Béri, Fermion-parity-based computation and its Majorana-zero-mode implementation, *Phys. Rev. Lett.* **128**, 180504 (2022).
- [64] S. Paesani and B. J. Brown, High-threshold quantum computing by fusing one-dimensional cluster states, *Phys. Rev. Lett.* **131**, 120603 (2023).
- [65] P. Krantz, M. Kjaergaard, F. Yan, T. P. Orlando, S. Gustavsson, and W. D. Oliver, A quantum engineer's guide to superconducting qubits, *Applied Physics Reviews* **6**, 021318 (2019).
- [66] C. D. Bruzewicz, J. Chiaverini, R. McConnell, and J. M. Sage, Trapped-ion quantum computing: Progress and challenges, *Applied Physics Reviews* **6**, 021314 (2019).
- [67] G. Burkard, T. D. Ladd, A. Pan, J. M. Nichol, and J. R. Petta, Semiconductor spin qubits, *Rev. Mod. Phys.* **95**, 025003 (2023).
- [68] K. Wintersperger, F. Dommert, T. Ehmer, A. Hoursanov, J. Klepsch, W. Mauerer, G. Reuber, T. Strohm, M. Yin, and S. Luber, Neutral atom quantum computing hardware: performance and end-user perspective, *EPJ Quantum Technology* **10**, 32 (2023).
- [69] F. Setiawan and C. McLauchlan, Stim circuits and data for “Tailoring Dynamical Codes for Biased Noise: The X3Z3 Floquet Code”, <https://doi.org/10.5281/zenodo.14258878>.
- [70] Google Quantum AI, Exponential suppression of bit or phase errors with cyclic error correction, *Nature* **595**, 383 (2021).
- [71] T. D. Ellison, J. Sullivan, and A. Dua, Floquet codes with a twist, [arXiv:2306.08027](https://arxiv.org/abs/2306.08027) (2023).
- [72] P. Aliferis and A. W. Cross, Subsystem fault tolerance with the Bacon-Shor code, *Phys. Rev. Lett.* **98**, 220502 (2007).
- [73] C. Gidney, Stim: a fast stabilizer circuit simulator, *Quantum* **5**, 497 (2021).
- [74] E. Dennis, A. Kitaev, A. Landahl, and J. Preskill, Topological quantum memory, *Journal of Mathematical Physics* **43**, 4452 (2002).
- [75] J. Edmonds, Maximum matching and a polyhedron with

- 0, 1-vertices, *Journal of research of the National Bureau of Standards B* **69**, 55 (1965).
- [76] J. Edmonds, Paths, trees, and flowers, *Canadian Journal of mathematics* **17**, 449 (1965).
- [77] A. G. Fowler, Minimum weight perfect matching of fault-tolerant topological quantum error correction in average $O(1)$ parallel time, *Quantum Information and Computation* **15**, 145 (2014).
- [78] C. Wang, J. Harrington, and J. Preskill, Confinement-Higgs transition in a disordered gauge theory and the accuracy threshold for quantum memory, *Annals of Physics* **303**, 31 (2003).
- [79] J. W. Harrington, *Analysis of quantum error-correcting codes: symplectic lattice codes and toric codes*, PhD thesis, California Institute of Technology, Pasadena, California (2004).
- [80] D. Aasen, J. Haah, Z. Li, and R. S. Mong, Measurement quantum cellular automata and anomalies in Floquet codes, arXiv:2304.01277 (2023).
- [81] S. Lloyd, Capacity of the noisy quantum channel, *Phys. Rev. A* **55**, 1613 (1997).
- [82] I. Devetak, The private classical capacity and quantum capacity of a quantum channel, *IEEE Transactions on Information Theory* **51**, 44 (2005).

Supplementary Information for “Tailoring Dynamical Codes for Biased Noise: The X^3Z^3 Floquet Code”

F. Setiawan^{1,*} and Campbell McLauchlan^{2,3,†}

¹Riverlane Research Inc., Cambridge, Massachusetts 02142, USA

²Centre for Engineered Quantum Systems, School of Physics,
The University of Sydney, Sydney, NSW 2006, Australia

³Riverlane, Cambridge, CB2 3BZ, UK

I. DETAILS OF HONEYCOMB AND CSS FLOQUET CODES

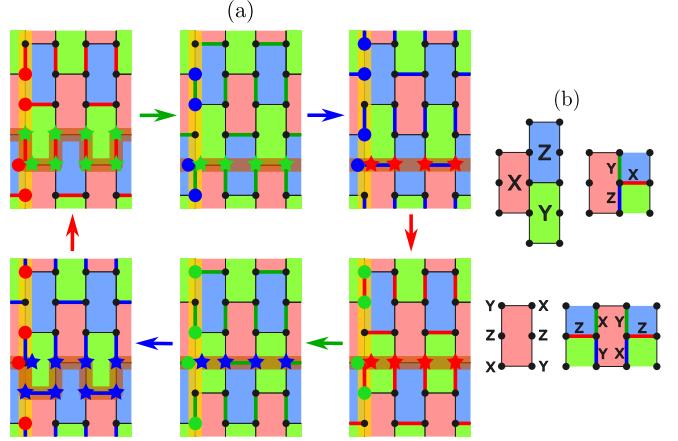
A. Honeycomb code

We here provide details of the two honeycomb code variants discussed in the main text. The codes’ logical operators evolve through their measurement cycles. In Supplementary Fig. 1(a), we show one set of logical operators evolving through two measurement cycles. While the measurement sequence has a period of 3, the logical operators only return to their initial values (up to signs) with a period of 6.

In Supplementary Fig. 1(b), we show the plaquette and edge operators for both honeycomb code variants. The P^6 code [1] has edge operators with Pauli bases defined by the edge colour, while the XYZ^2 code [2] has edge operators with bases defined by the edge’s orientation in the T junction, i.e., horizontal for the Z edges, and rotated 90° clockwise and counter-clockwise from the horizontal edges, for the X and Y edges, respectively [see Supplementary Fig. 1(b)]. The plaquette operators of the P^6 code also differ between plaquettes of different colours, while the plaquette operators of the XYZ^2 code are all the same. Each plaquette operator is a product of the edge operators along the plaquette boundary. For example, a red plaquette operator is the product of the green and blue edge operators around its boundary. The plaquette operators are measured in two subrounds. For instance, after measuring the g and b checks sequentially, we can learn the value of each red plaquette by multiplying the measurement values of the green and blue checks around its boundary. Unlike the CSS and X^3Z^3 Floquet codes, the honeycomb code has constant plaquette stabilisers.

B. CSS Floquet code

In Supplementary Fig. 2, we illustrate the measurement cycle of the CSS Floquet code [3, 4] and the resulting evolution of (one set of) its logical operators. Its logical operators are either of Z -type or X -type for all time, as shown. We show a Z -type logical operator defined



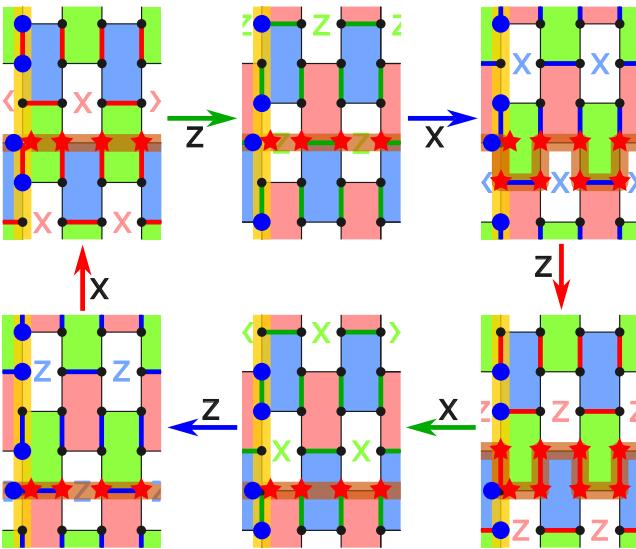
Supplementary Figure 1. The honeycomb code measurement cycle, logical operators, plaquettes and edges. (a) A distance-4 (under code-capacity noise) P^6 code [1] is shown throughout its evolution. The type of check measured at each measurement subround is indicated by the arrows and the highlighted/coloured edges in the lattice. One set of anti-commuting logical operators is shown in yellow and brown-coloured paths, where their qubit supports are depicted using big circles and stars, respectively. Each logical is a product of Pauli operators on the coloured qubits along its cycle, where red, green and blue qubits support X , Y , and Z Pauli bases, respectively. The other set of logical operators is offset by three measurement subrounds. (b) Top: Red, green and blue edges/plaquettes of the P^6 honeycomb code support X , Y and Z operators, respectively. Bottom: All plaquettes of the XYZ^2 honeycomb code support the same XYZ^2 operator, as shown on the left. Edge operators are defined based on their orientation in the T junction (rather than the edge colour), as shown on the right.

on a vertical non-trivial cycle of the lattice. However, there exists another X -type logical operator (not shown) defined on that same vertical cycle. One can find this operator (and its anti-commuting partner) by noting the symmetry in the code obtained by proceeding forward by three measurement subrounds ($t \mapsto t + 3$) and applying a Hadamard gate to all qubits ($X \leftrightarrow Z$).

The plaquette operator members of the ISG at each step of the measurement cycle are indicated in Supplementary Fig. 2. The value of each plaquette operator is inferred, in one time step, from the product of the measurement values of three edges surrounding the plaquette. At each measurement subround, only one kind

* setiawan.wenming@gmail.com

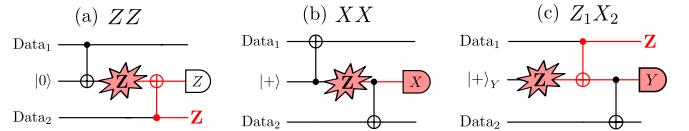
† campbell.mclauchlan@gmail.com



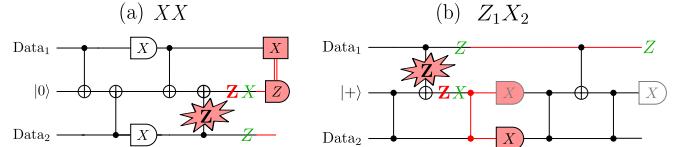
Supplementary Figure 2. **Measurement cycle and logical operator evolution for the CSS Floquet code.** X or Z edge operators are measured alternately, with the colour of edge and Pauli type indicated by the labelled arrows. Plaquettes labelled with X or Z labels host only a stabiliser operator of the corresponding type, while other plaquettes host both X - and Z -type stabilisers. One set of anti-commuting logical operators is shown by yellow and brown paths, where their qubit supports are depicted using big circles and stars, respectively. Each logical is a product of only one type of Pauli operators, either X or Z , represented by the red or blue coloured qubits.

[which is of one type (X or Z) and one colour] of plaquette is reinitialised and one other kind is re-measured to form a detector. The reinitialised and re-measured plaquettes have the same (X or Z) type as the checks performed at that particular subround, but are of different colours from each other and also from the checks [3, 4]. The colours of the reinitialised and re-measured plaquettes are respectively the same as the colours of the checks performed just before and immediately after the current subround.

To see how detectors are formed in this code, let us follow a single blue X plaquette through the measurement sequence. After red X checks (top left of Supplementary Fig. 2), the blue X plaquette operator is measured (we infer its outcome by multiplying together the check measurement outcomes around its boundary). This initialises the plaquette. It is only re-measured after the green X check measurements, four subrounds later, at which point a detector is formed. After this, blue Z checks are measured, which anti-commute with the blue X plaquette operator. Hence, its measurement outcome after this subround is random.



Supplementary Figure 3. **Error propagation in the depth-2 parity-check circuits of Fig. 4 in the main text.** Shown are the examples of Z errors occurring at certain space-time points of (a) ZZ , (b) XX , and (c) Z_1X_2 parity check circuits. These errors can spread to at most single-qubit Z errors on the data qubits and flip certain measurement outcomes (shown in red).



Supplementary Figure 4. **Error propagation in the parity check circuits of Fig. 5 in the main text.** Shown are examples of Z errors occurring on the target qubits during the application of a CNOT gate in two different parity-check circuits: (a) XX and (b) Z_1X_2 . Green Paulis represent Pauli errors occurring in a specific branch of the state vector, while red Paulis occur in all branches. Red lines track the spread of Pauli errors and red measurements may be flipped by the errors.

II. PARITY-CHECK MEASUREMENT CIRCUITS

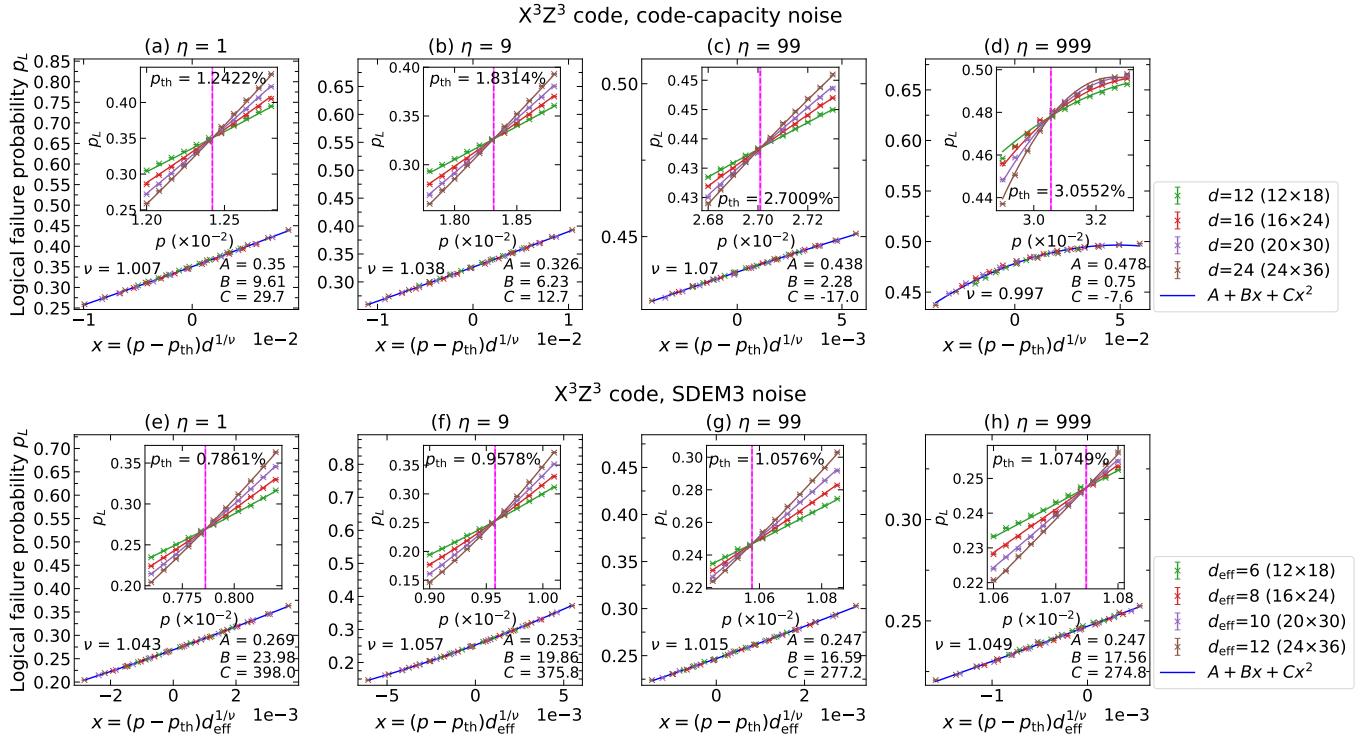
A. Error propagation

Here we provide more details on the propagation of errors, including mid-gate errors, in the parity-check circuits shown in Figs. 4 and 5 of the main text. We first assume there are no mid-gate errors, and demonstrate that the parity check circuits in Fig. 4 of the main text is bias-preserving. To this end, in Supplementary Fig. 3 we provide examples of Z errors on ancilla qubits and show that these errors can only spread to the data qubit as a Z error; thus, the strength of the noise bias is not degraded.

When considering mid-gate errors, the analysis is more involved. To see how the mid-gate errors propagate, we consider a CNOT gate implemented via the following Hamiltonian [5]:

$$H_{\text{CX}} = V \frac{1}{2} [(\mathbb{1}_c + Z_c) \otimes \mathbb{1}_t + (\mathbb{1}_c - Z_c) \otimes X_t], \quad (1)$$

where V is the interaction strength between the control c and target qubit t . This Hamiltonian gives rise to a unitary gate $U(T) = \exp(-iT H_{\text{CX}})$ at time T where for $VT = \pi/2$, the resulting unitary is a CNOT gate. Note that a phase-flip error Z_c , occurring on the control qubit c during the application of the CNOT gate, always propagate as a Z error since it commutes with the Hamiltonian



Supplementary Figure 5. **Logical failure probability p_L of the X^3Z^3 Floquet code as a function of $x = (p - p_{\text{th}})d^{1/\nu}$ or $x = (p - p_{\text{th}})d_{\text{eff}}^{1/\nu}$.** They are calculated for two different noise models: Code capacity (upper panels) and SDEM3 (lower panels), with various bias strengths: (a,e) $\eta = 1$, (b,f) $\eta = 9$, (c,g) $\eta = 99$, and (d,h) $\eta = 999$. We fit the results to the function $A + Bx + Cx^2$ (blue line). Insets: Logical failure probability p_L vs physical error rate p ; thresholds p_{th} are shown by the magenta vertical lines. Results for each distance d and physical error rate p are averaged over $10^6 - 10^7$ number of shots.

H_{CX} [Supplementary Eq. (1)]. On the other hand, a mid-gate phase flip error Z_t on the target qubit can propagate as a phase flip on the control qubit and a combination of phase flip and bit flip error on the same target qubit, as shown below. When the mid-gate error Z_t happens at time $0 < \tau < T$, the resulting unitary rotation of the two qubits is:

$$\begin{aligned} U(T - \tau)Z_tU(\tau) &= Z_t e^{iV(T-\tau)(\mathbb{1}_c - Z_c)X_t} U(T) \\ &= Z_t \left(\cos^2 \theta + i \frac{1}{2} \sin 2\theta (X_t - Z_c X_t) + \sin^2 \theta Z_c \right) U(T), \end{aligned} \quad (2)$$

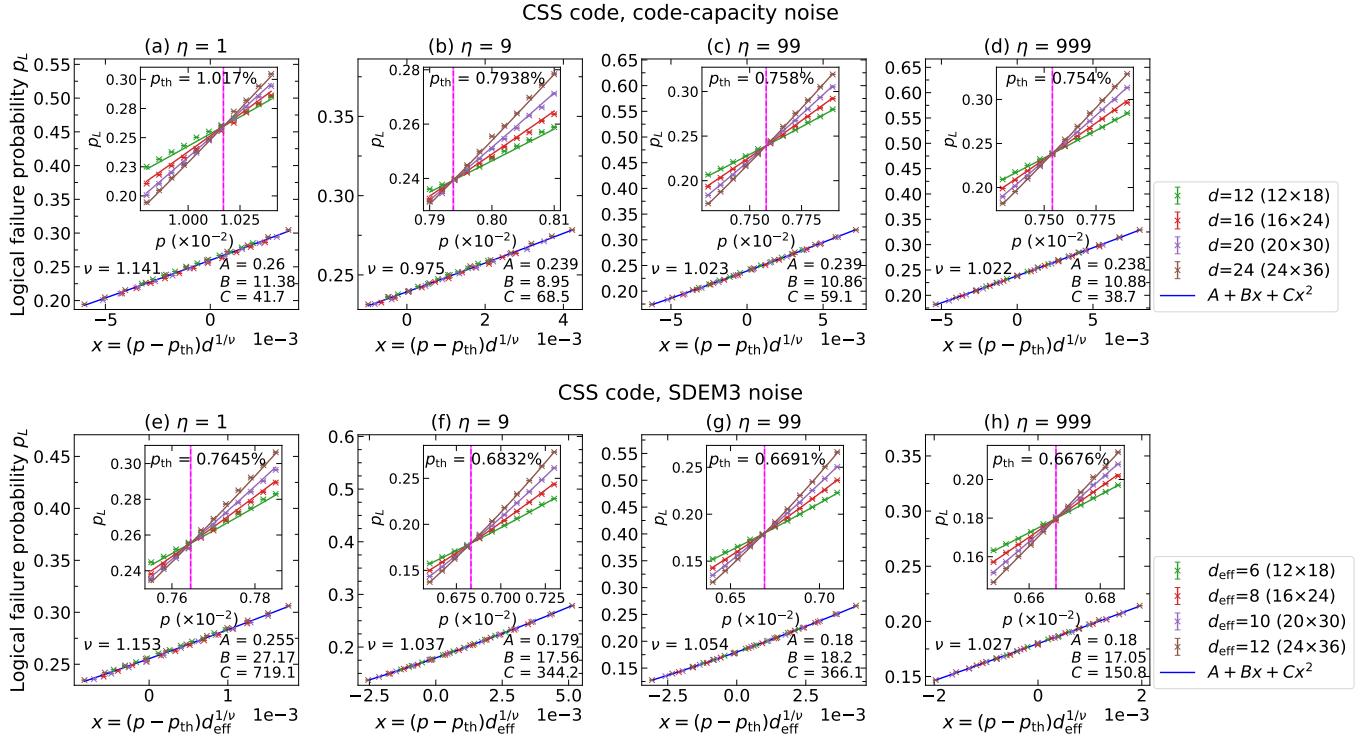
where $\theta \equiv V(T - \tau)$. The unitary error $\exp[iV(T - \tau)(\mathbb{1}_c - Z_c)X_t]$ in Supplementary Eq. (2) is an X rotation on the target qubit t controlled by qubit c . As shown by the terms in the parentheses in the last line of Supplementary Eq. (2), the resulting unitary error is as a superposition of different Pauli errors: $\mathbb{1}$, X_t , Z_c and $Z_c X_t$.

We now consider specifically the error shown in Supplementary Fig. 4(a), occurring during the final CNOT gate. The green Paulis represent errors occurring in a particular branch of the wavefunction. The red \mathbf{Z} operator is the Pauli error that exists for all branches. As can be seen, the X_a error on ancilla qubit a flips the Z_a

measurement on the ancilla. Since the ancilla is used as the control qubit of the subsequent classically controlled X gate on data qubit 1, this Z_a measurement flip results in an incorrectly applied X_1 gate on data qubit 1. Thus, a mid-gate Z_a error can propagate to an X_1 error. In contrast, consider the mid-gate error shown in Supplementary Fig. 4(b). The error may flip some measurement outcomes (shown in red), but the resulting Pauli errors after the circuit is simply a superposition of $\mathbb{1}$ and Z errors on one of the data qubits. Hence, the noise bias is not degraded.

B. Bias preservation of XX parity-check circuit

The XX parity-check circuit needs to contain at least one two-qubit gate which does not preserve the Z bias on the data qubit. This seems to be a necessary feature of an XX measurement circuit. To see this, note that (as described in the Results section of the main text) the value of $X_1 X_2$ can be obtained through a single measurement (whose result is the outcome of the $X_1 X_2$ measurement) or by measuring the $X_1 P_a$ and $X_2 P_a$ operators, where P_a is the Pauli operator on some ancillary degree of freedom. In the former case, we require CNOT gates targeting data qubits [see Fig. 4(b) of the main text for the



Supplementary Figure 6. **Logical failure probability p_L of the CSS Floquet code as a function of $x = (p - p_{\text{th}})d^{1/\nu}$ or $x = (p - p_{\text{th}})d_{\text{eff}}^{1/\nu}$.** They are calculated for two different noise models: Code capacity (upper panels) and SDEM3 (lower panels), with various bias strengths: (a,e) $\eta = 1$, (b,f) $\eta = 9$, (c,g) $\eta = 99$, and (d,h) $\eta = 999$. We fit the results to the function $A + Bx + Cx^2$ (blue line). Insets: Logical failure probability p_L vs physical error rate p ; thresholds p_{th} are shown by the magenta vertical lines. Results for each distance d and physical error rate p are averaged over 10^6 number of shots.

case where we measure the ancilla qubit and for the case where data qubit is measured, we need long-range CNOT gates between the data qubits analogous to Fig. 5(c) of the main text], which degrade the Z bias on the target data qubits. In the latter case, we need to use a non-bias preserving classically conditioned X gate on one of the data qubits after we perform measurement on the data qubits to obtain the $X_1 P_a$ and $X_2 P_a$ parity values. In order for the $X_1 P_a$ and $X_2 P_a$ measurements not to reveal the values of X_1 and X_2 separately, the ancilla is prepared and later measured in the eigenstate of \bar{P}_a , some Pauli basis which anticommutes with P_a . In Fig. 5(b) of the main text, we choose $P_a = X_a$ and $\bar{P}_a = Z_a$. The ancilla measurement result is then used to perform the classically controlled X gate on one of the data qubits.

To see why a classically controlled gate is required in the XX parity check circuit in Fig. 5(b) of the main text, we can follow the evolution of the wavefunction of the data qubits and ancilla after the $X_1 X_a$ and $X_2 X_a$ measurements. We consider the ancilla to be prepared in the state $|0\rangle$ and the two data qubits are initialised in the state

$$|\psi\rangle = \alpha|00\rangle_{1,2} + \beta|01\rangle_{1,2} + \gamma|10\rangle_{1,2} + \delta|11\rangle_{1,2}. \quad (3)$$

After the $X_1 X_a$ and $X_2 X_a$ measurements, the wavefunc-

tion evolves as

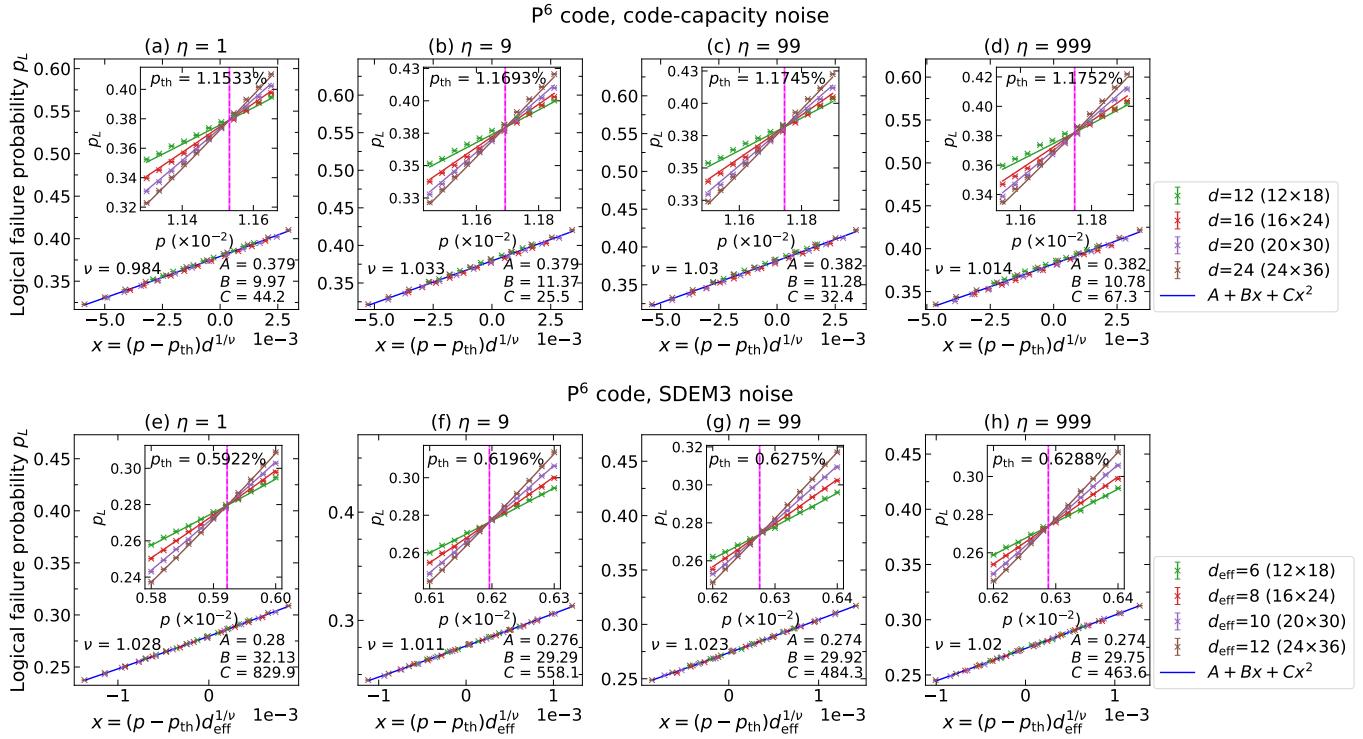
$$\begin{aligned} |\psi\rangle |0\rangle & \xrightarrow{X_1 X_a} \frac{1}{\sqrt{2}} (|\psi\rangle |0\rangle_a + (-1)^{m_{X_1 X_a}} X_1 |\psi\rangle |1\rangle_a) \\ & \xrightarrow{X_2 X_a} \frac{1}{2} \left\{ (|\psi\rangle + (-1)^{m_{X_1 X_a} + m_{X_2 X_a}} X_1 X_2 |\psi\rangle) |0\rangle_a \right. \\ & \quad \left. + [(-1)^{m_{X_1 X_a}} X_1 |\psi\rangle + (-1)^{m_{X_2 X_a}} X_2 |\psi\rangle] |1\rangle_a \right\} \\ & = \frac{1}{\sqrt{2}} \left(|\tilde{\psi}\rangle_{X_1 X_2} |0\rangle_a + (-1)^{m_{X_1 X_a}} X_1 |\tilde{\psi}\rangle_{X_1 X_2} |1\rangle_a \right), \end{aligned} \quad (4)$$

where

$$|\tilde{\psi}\rangle_{X_1 X_2} \equiv \frac{1}{\sqrt{2}} [|\psi\rangle + (-1)^{m_{X_1 X_2}} X_1 X_2 |\psi\rangle], \quad (5)$$

is the eigenvector of $X_1 X_2$ with eigenvalue $m_{X_1 X_2} = 0, 1$. In evaluating the last line of Supplementary Eq. (4), we have used the relation $(-1)^{m_{X_1 X_2}} = (-1)^{m_{X_1 X_a} + m_{X_2 X_a}}$ since $(X_1 X_a) \cdot (X_2 X_a) = X_1 X_2$, where $m_{X_1 X_a}, m_{X_2 X_a} = 0, 1$ are the $X_1 X_a$ and $X_2 X_a$ measurement results, respectively.

As can be seen from last line of Supplementary Eq. (4), the state of the data qubits after the $X_1 X_a$ and $X_2 X_a$ measurements is entangled with the ancilla qubit's state.



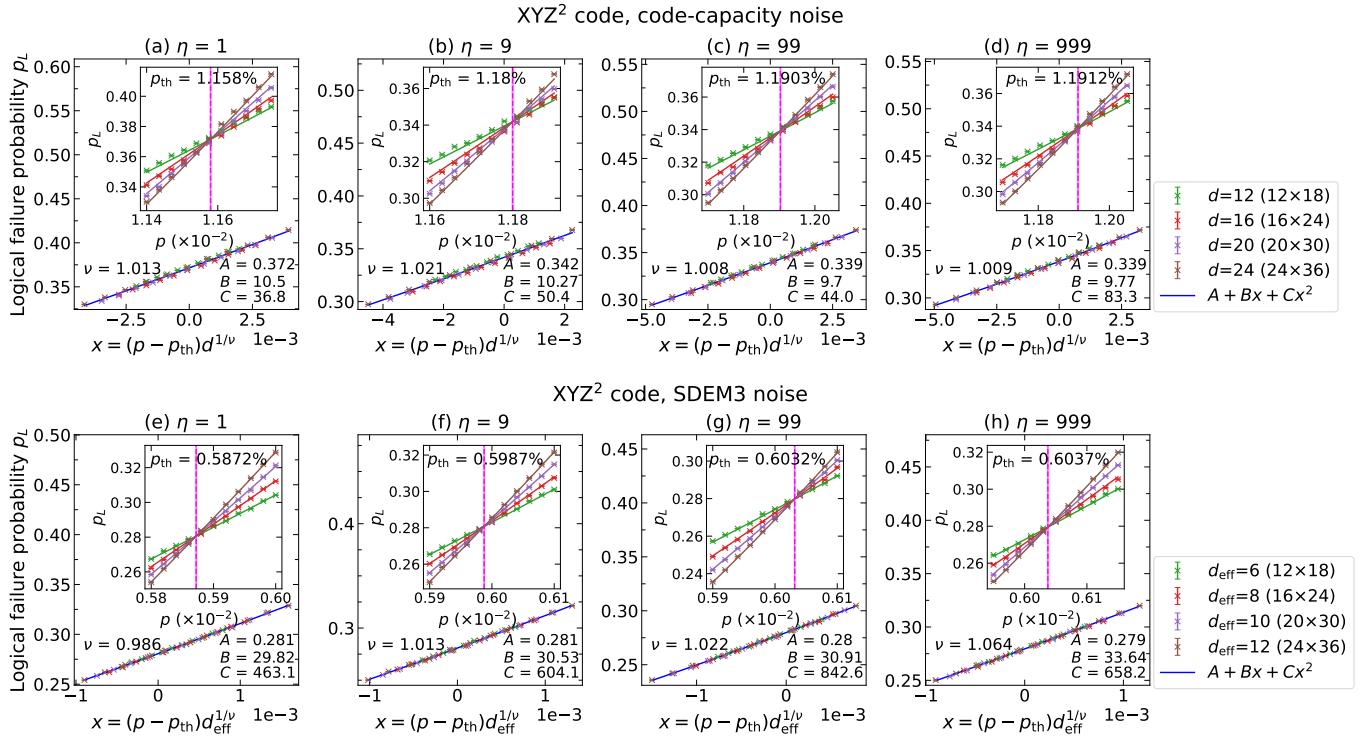
Supplementary Figure 7. **Logical failure probability p_L of the P^6 honeycomb code as a function of $x = (p - p_{\text{th}})d^{1/\nu}$ or $x = (p - p_{\text{th}})d_{\text{eff}}^{1/\nu}$.** They are calculated for two different noise models: Code capacity (upper panels) and SDEM3 (lower panels), with various bias strengths: (a,e) $\eta = 1$, (b,f) $\eta = 9$, (c,g) $\eta = 99$, and (d,h) $\eta = 999$. We fit the results to the function $A + Bx + Cx^2$ (blue line). Insets: Logical failure probability p_L vs physical error rate p ; thresholds p_{th} are shown by the magenta vertical lines. Results for each distance d and physical error rate p are averaged over 10^6 number of shots.

Depending on the ancilla measurement result, we will obtain two different data-qubit states where these two states differ by a Pauli X operator acting on one of the data qubits and a trivial global phase. As a result, we need an X gate acting on one of the data qubits classically conditioned on the ancilla measurement to cancel this residual X_1 operator. This gate, however, does not preserve the noise bias on the target data qubit [see Supplementary Fig. 4(a)]. In conclusion, this observation leads us to believe that constructing fully bias-preserving XX check circuit using conventional two-qubit gates is impossible.

C. Circuit-depth optimality

Finally, we discuss the circuit-depth optimality of these circuits. If we assume that bias-preserving two-qubit gates [6] are available, it is clear that the depth-2 parity check circuits in Fig. 4 of the main text are optimal for *weight-two* parity checks to be bias-preserving. Next, for the case where the bias-preserving parity check circuits are constructed using conventional two-qubit gates, we will argue that the parity check circuits shown in Fig. 5 of the main text are also already optimal in circuit depth, where further reduction in the depth of the

circuits may not be possible without degrading the noise-bias preservation of the circuit. First of all, the ZZ circuit of Fig. 5(a) in the main text and Z_1X_2 circuit on the left-hand side of Fig. 5(c) of the main text (the latter of which assumes the availability of direct connections between data qubits) have a circuit depth of 2 which is optimal for *weight-two* parity checks. For the case where long-range connectivity between data qubits is not available, we now argue that the depth-4 circuit shown in the last equality of Fig. 5(c) of the main text is a shortest-depth implementation of the circuit on the left-hand side of Fig. 5(c) of the main text. The argument is as follows. We first note that we cannot have a *bias-preserving* Z_1X_2 measurement by measuring the ancilla qubit connecting the two data qubits. This is because such a circuit would require a CNOT gate targeting data qubit 2 [as in Fig. 4(c) of the main text], which does not preserve the noise bias. As a result, the bias-preserving Z_1X_2 measurement needs to be constructed as in Fig. 5(c) of the main text, where we measure one of the data qubits after applying nearest-neighbour two-qubit gates between the ancilla and data-qubits in a bias-preserving way, i.e., CNOT gates targeting ancilla qubits or CZ gates. Since at least 2 nearest-neighbour gates are required to implement one next-nearest neighbour two-qubit gate and the Z_1X_2 circuit requires two such next-nearest neighbour



Supplementary Figure 8. Logical failure probability p_L of the XYZ^2 honeycomb code as a function of $x = (p - p_{\text{th}})d^{1/\nu}$ or $x = (p - p_{\text{th}})d_{\text{eff}}^{1/\nu}$. They are calculated for two different noise models: Code capacity (upper panels) and SDEM3 (lower panels), with various bias strengths: (a,e) $\eta = 1$, (b,f) $\eta = 9$, (c,g) $\eta = 99$, and (d,h) $\eta = 999$. We fit the results to the function $A + Bx + Cx^2$ (blue line). Insets: Logical failure probability p_L vs physical error rate p ; thresholds p_{th} are shown by the magenta vertical lines. Results for each distance d and physical error rate p are averaged over 10^6 number of shots.

gates, the depth-4 Z_1X_2 circuit on the right-hand side of the last equality in Fig. 5(c) of the main text is therefore optimal.

We now comment on the tradeoff between using the parity-check circuits in Fig. 4 and Fig. 5 of the main text, for the case where bias-preserving two-qubit gates are unavailable. Compared to the parity check circuits in Fig. 4 of the main text, the circuits presented in Figs. 5(b) and 5(c) of the main text have better mid-gate error bias-preserving quality at the expense of additional gate depth. This trade-off will only be beneficial if the adverse effect of the extra noise due to longer gate depth is outweighed by the beneficial bias-preserving quality of the circuit, which is expected to be true when the noise bias is sufficiently large. To justify this, one would therefore need to compare the performance of the bias-preserving and non-bias preserving parity check circuits by implementing them experimentally or simulating them using noise parameters specific to the device considered. We leave this study to future works.

III. THRESHOLD CALCULATIONS

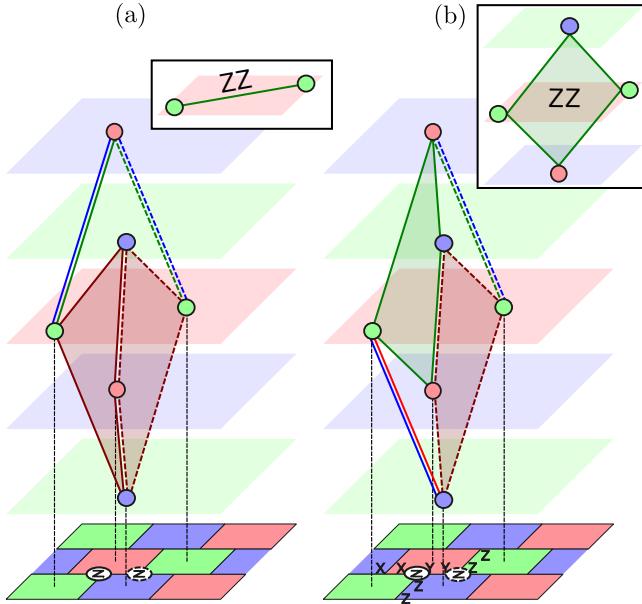
Supplementary Figures 5–8 show the resulting data collapse and close-to-threshold plots of the logical failure

probabilities for the X^3Z^3 , CSS, P^6 and XYZ^2 Floquet codes, respectively. The plots for each code are calculated for various bias strengths. As described in the Methods section of the main text, the thresholds are obtained by fitting the logical failure probability p_L data taken for various physical error rates p and code distances to the function $A + Bx + Cx^2$, where $x = (p - p_{\text{th}})d^{1/\nu}$ or $x = (p - p_{\text{th}})d_{\text{eff}}^{1/\nu}$ for the code-capacity and SDEM3 noise models, respectively. Here, A, B, C are the fit parameters, ν is the critical exponent, p_{th} is the threshold, d and d_{eff} are code distances for the code-capacity and SDEM3 noise models, respectively.

IV. HYPEREDGE ERRORS IN HONEYCOMB CODES UNDER INFINITE NOISE BIAS

In this section we explain the difference in performance between the XYZ^2 and P^6 codes. Specifically, we focus on the hyperedges formed in the detector hypergraphs under an infinitely phase-biased SDEM3 noise model. We will show that the frequency of hyperedges due to two-qubit errors is greater for the XYZ^2 code, resulting in an inferior performance under the biased SDEM3 noise, compared to the P^6 code.

Unlike the CSS and X^3Z^3 Floquet codes, the single-



Supplementary Figure 9. Detector hyperedges induced by Z errors in honeycomb codes. Shown are examples of detector hypergraphs of (a) P^6 and (b) XYZ^2 honeycomb codes. Coloured horizontal layers indicate type of measurements performed. Coloured nodes are detectors. Hyperedges and edges between detectors correspond to the syndromes of the single-qubit Z errors shown by the uncoloured circles. Solid (dashed) edges correspond to syndromes due to the Z error on the left (right) qubit. The colour of the (hyper)edge denotes the time step during which the Z error occurs, e.g., a red hyperedge is the syndrome for a Z error occurring immediately after red check measurements. Insets show the syndromes for the two-qubit ZZ error occurring on the two qubits indicated (along a green edge) immediately after green check measurements. This syndrome is edge-like (hyperedge-like) for the P^6 (XYZ^2) code.

qubit X and Z Pauli errors in both of the honeycomb code variants can produce hyperedge syndromes [7, 8] if they occur at specific measurement subrounds. These hyperedges degrade the performance of a matching decoder as they must be decomposed into edges, thereby losing some information about the correlations between detectors. We show in Supplementary Fig. 9 an example of such hyperedges which are formed from single-qubit Z errors. Note that for a single-qubit Z error to produce a hyperedge-like syndrome, it must occur between the measurements of the XX and YY edges of the Z -error affected qubit.

First, let us consider the P^6 honeycomb code. Since in the P^6 code, the edge type is determined by its colour, a single-qubit Z error here generates a hyperedge syndrome (four flipped detectors) if it occurs between red (XX) and green (YY) check measurements [see the hyperedge highlighted in red in Supplementary Fig. 9(a)] and an edge-like syndrome (two flipped detectors) if it occurs at other measurement subrounds [shown by blue/green edges in Supplementary Fig. 9(a)]. Notice that the product of

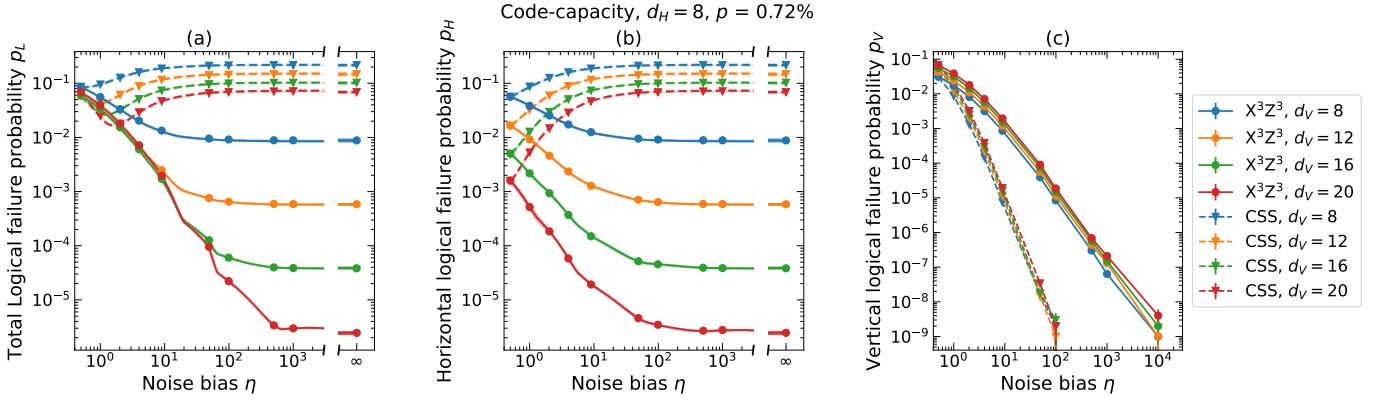
the two Z errors occurring simultaneously always results in a graph-like syndrome whenever the error occurs [see inset of Supplementary Fig. 9(a)]. This is the same for all neighbouring ZZ errors occurring along edges in the lattice.

In the XYZ^2 honeycomb code, by contrast, hyperedges can be formed from Z errors occurring at every measurement subround since edges of each colour consist of all check types (XX , YY and ZZ). The hyperedge-forming Z errors, however, need to occur at specific locations such that they happen between the measurements of the XX and YY edges of the noisy qubits. That is, a Z error with coordinates (x, y, t) could form a hyperedge syndrome for any t , given suitably chosen x and y . Two such Z errors and their syndromes for different time steps are shown in Supplementary Fig. 9(b). As can be seen, one Z error (the error on the right) produces a hyperedge if it occurs after red check measurements, while the other (the error on the left) does so only after the green measurement subround. As such, a neighbouring ZZ error occurring at a given time always produces a hyperedge syndrome. For example, if the ZZ error shown (supported by a green edge) in Supplementary Fig. 9(b) occurs after the green MPP measurements, the resulting syndrome will include four detectors [see inset of Supplementary Fig. 9(b)].

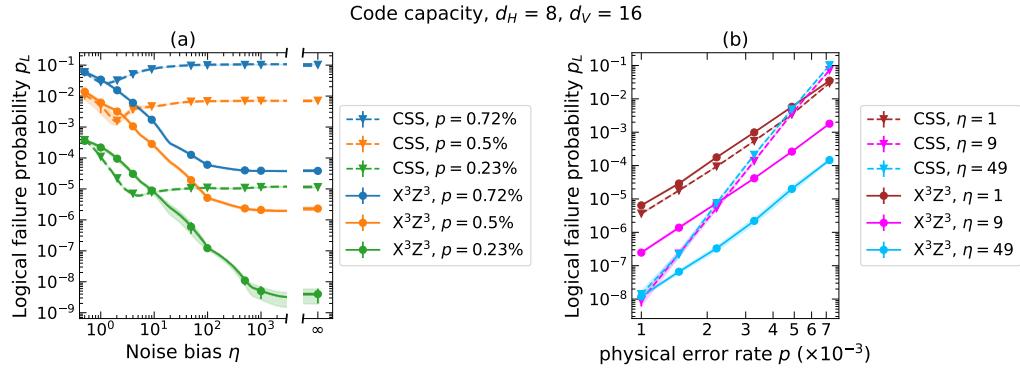
In the SDEM3 noise model, two-qubit ZZ errors along an edge can occur only after the measurement of the corresponding edge operator. For example, a ZZ error along a green edge can only occur after the green measurement subround. We have shown that such errors always produce edge-like syndromes for the P^6 code and hyperedge-like syndromes for the XYZ^2 code. On the other hand, the number of possible hyperedge-like syndromes from single-qubit Z errors and measurement errors is the same for both codes. As a result, the total number of hyperedges is smaller for the detector hypergraph of the P^6 code, resulting in a better code performance.

V. PERFORMANCE OF ELONGATED CSS AND X^3Z^3 FLOQUET CODES

Supplementary Figure 10 shows the subthreshold performance of elongated CSS and X^3Z^3 Floquet codes with different aspect ratios d_V/d_H . The code distances d_H and d_V are defined as the minimum weight of the code's logical operators in the horizontal and vertical directions, respectively. The total logical failure probabilities plotted in Supplementary Fig. 10(a) are calculated using Eq. (9) in the main text, from the horizontal and vertical logical error probabilities shown in Supplementary Figs. 10(b) and (c), respectively. As shown in Supplementary Fig. 10(a), both the CSS and X^3Z^3 Floquet codes have their logical errors improve with increasing d_V . Even though for a fixed lattice size, the CSS and X^3Z^3 Floquet codes have the same the horizontal and vertical code distances, these two codes have different performance under biased noise. In the noise regime



Supplementary Figure 10. **Subthreshold performance of elongated CSS and X^3Z^3 Floquet codes.** They are plotted as a function of noise bias η for different vertical distances d_V . (a) Total logical failure probability p_L [Eq. (9) of the main text] calculated from (b) Horizontal logical failure probability p_H , and (c) Vertical logical failure probability p_V . Results are simulated under the code-capacity noise model for a fixed physical error rate $p = 0.72\%$ and horizontal distance $d_H = 8$. Each data point is averaged over $10^6 - 10^{10}$ number of shots.



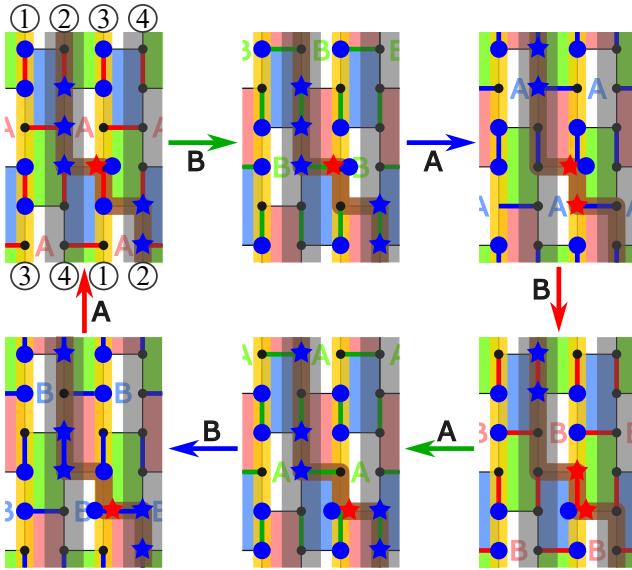
Supplementary Figure 11. **Logical failure probability of elongated CSS and X^3Z^3 Floquet codes.** They are plotted as a function of (a) noise bias η , and (b) physical error rate p . Results are calculated for the code-capacity noise model with code distances $d_H = 8$ and $d_V = 16$. Each data point is averaged over $10^8 - 10^9$ number of shots.

near depolarising noise (and particularly for lower physical rate; see Supplementary Fig. 11), the logical error is slightly lower for the CSS code. On the other hand, as the noise becomes biased towards dephasing error, the X^3Z^3 code's logical error becomes increasingly better and can reach several orders of magnitude lower than those of the CSS Floquet code.

The performance difference of these two codes in the two different noise regimes is due to the fact that for highly rectangular patches in the highly dephasing (depolarising) noise regime, the total logical error probability is dominated by the horizontal (vertical) logical failure probability, p_H (p_V). For large η , p_H is lower for the X^3Z^3 code, while for small η , p_V is similar between the two codes [see Supplementary Figs. 10(b) and (c)]. Indeed, Supplementary Fig. 10(b) shows that as the noise bias increases, the horizontal logical error probability of the CSS code increases while that of the X^3Z^3 code decreases. The reason for this is that all the vertical logical operators in the CSS code are of pure Pauli Zs while half of the vertical logical strings for the X^3Z^3 code are of pure

X -type. Therefore, as the Z noise bias increases, the probability of the errors forming vertical logical strings that flip the horizontal logical observables increases for the CSS code while it decreases for the X^3Z^3 code. Furthermore, the X^3Z^3 code has a much simpler decoding problem, as explained in the main text.

Supplementary Figure 10(c) shows that as the Z noise bias increases, the vertical logical error probability of the CSS code decreases much faster than that of the X^3Z^3 code. This is because the CSS code's horizontal logical operators, which flip the vertical logical observable, consist of only X Paulis, and these pure X -horizontal operators become exceedingly less likely to form as the Z noise bias increases. Meanwhile, each of the X^3Z^3 code's horizontal logicals comprise both X and Z Paulis. As a result, under Z -biased noise the probability of errors forming the horizontal logical strings that flip the Z -logical observables is lower for the CSS code. However, owing to $p_H \gg p_V$ for large η , the combined logical failure probability p_L for the X^3Z^3 code is much lower than the CSS code, as can be seen in Supplementary Fig. 10(a).

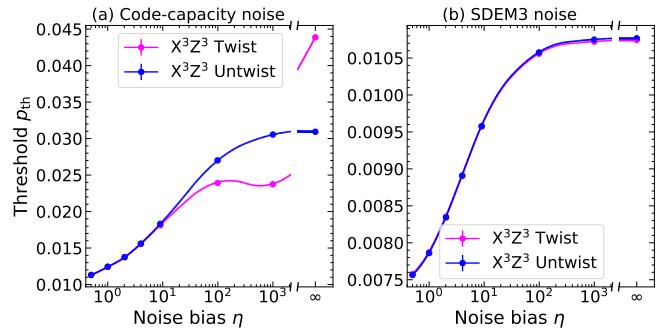


Supplementary Figure 12. Evolution of logical operators in the X^3Z^3 Floquet code with a twisted boundary condition. Number-labelled edges at the top and bottom boundaries indicate the twisted boundary condition. The logical operators are shown in yellow and brown non-trivial cycles, where their qubit supports are depicted using big circles and stars, respectively. Each logical is a product of Pauli operators on the coloured qubits along its cycle, where red and blue qubits support X , and Z Pauli bases, respectively.

While our results are simulated for the periodic boundary case, the above conclusion is expected to hold also for the open boundary case. This is because the X^3Z^3 Floquet code with open boundary conditions has the same domains aligned in the same direction (i.e., the vertical shaded and unshaded strips) as in the periodic boundary condition case. As a result, pure Z -type logical operators are aligned along the elongated direction while mixed X/Z operators lie in the transverse direction, just as is the case with the unrotated XZZX code [see Fig. 1(h) in Supplementary Ref. [9]]. Since changing the boundaries does not change the Pauli supports of the logical operators in the X^3Z^3 Floquet code, the likelihood of the errors forming logical string operators for the open boundary case will be similar to that of the periodic boundary case. This is in contrast to the rotated XZZX code where domains lie diagonally across the patch. As shown in Supplementary Ref. [10], when a rectangular lattice is applied to a rotated XZZX code, it has a shorter Z-code distance which leads to a performance degradation.

VI. X^3Z^3 FLOQUET CODE WITH TWISTED BOUNDARY CONDITIONS

In this section, we present simulation results obtained for the X^3Z^3 code with a twisted boundary condition. The twisted boundary refers to the case where the qubits

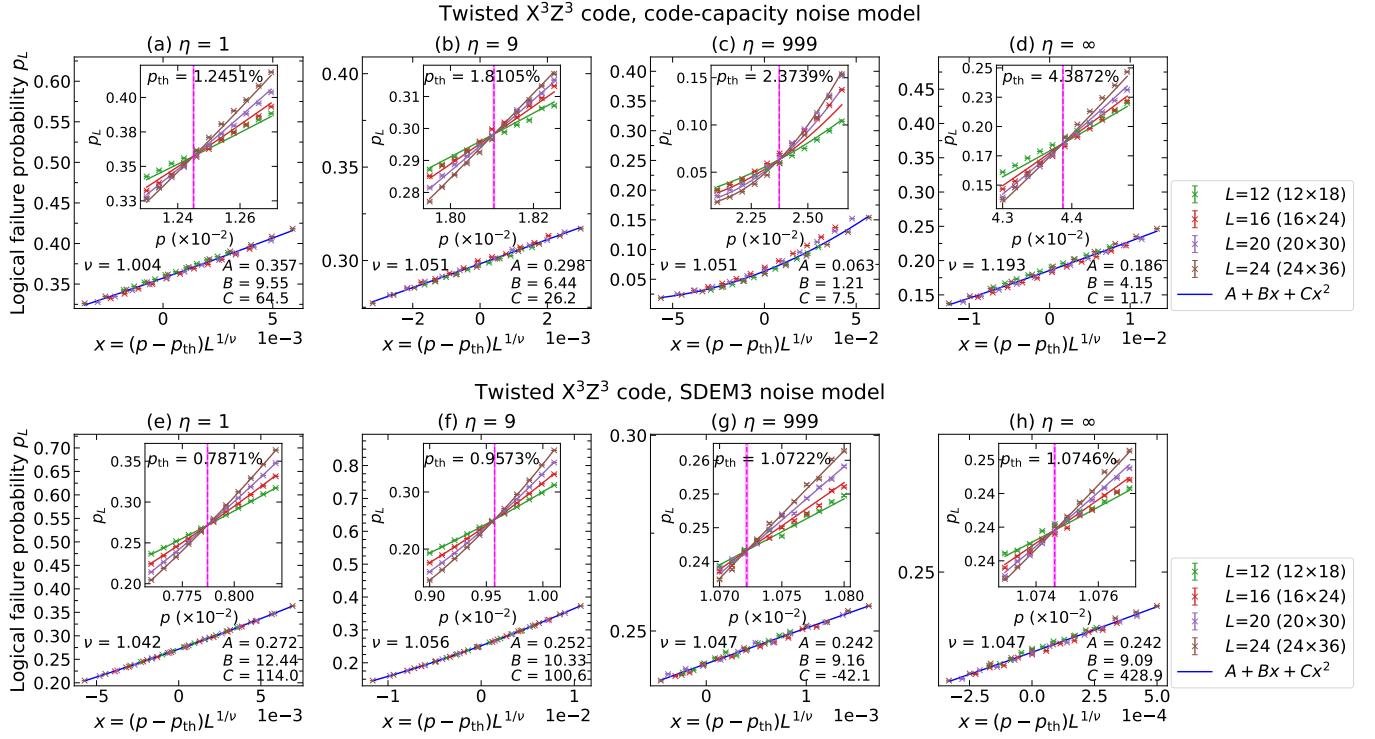


Supplementary Figure 13. Thresholds (p_{th}) of the X^3Z^3 Floquet codes with different periodic boundary conditions. Thresholds for twisted and untwisted periodic boundary conditions are shown in magenta and blue, respectively. They are plotted as a function of noise bias η for two different noise models: (a) code-capacity and (b) SDEM3. We show some of the data points used in obtaining the thresholds in Fig. 14. For better visualisation, we fit all curves with quadratic splines.

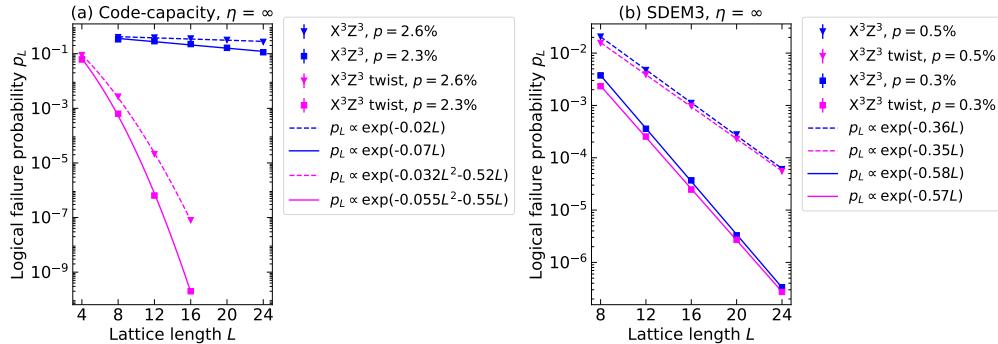
at the top row of the honeycomb lattice are connected to qubits at the bottom row, shifted by two columns (see Supplementary Fig. 12). In the figure, we also show two (of the four) logical operators. The yellow logical operator, which consists of only Z operators, has a length $L^2/2$ for a lattice with a size $L \times 3L/2$. This is longer than the length of the untwisted code logical operator, which is only L (or $2L$) [cf. Fig. 1(c) of the main text]. As a result, the performance of the twisted code near infinite bias is expected to be better than the untwisted code since it will be harder for the pure dephasing errors to form such a long logical operator.

To support this argument, we present the thresholds of such a code in Supplementary Fig. 13, where we compare them to the thresholds of the untwisted code. Data used to obtain the thresholds are presented in Supplementary Fig. 14. We find that there is a stronger finite-size effect for the twisted code compared to the untwisted one as can be seen from the finite-size collapse plots. As shown in Supplementary Fig. 13, the twisted code has better thresholds than the untwisted code only in the regime close to infinite bias. For the small noise bias regime ($\eta \lesssim 10$), the thresholds of the twisted code are roughly similar to those of the untwisted code. However, in the intermediate bias regime, the thresholds of the twisted code are lower than those of the untwisted code.

Besides the thresholds, we also compute the logical failure probability of the twisted X^3Z^3 Floquet code as a function of the lattice length L . As can be seen from Supplementary Fig. 15, at infinitely phase-biased code-capacity noise the logical failure probability of the untwisted code decays as $p_L \propto \exp(-\gamma_1 L)$, whereas that of the twisted code decays as $p_L \propto \exp(-\gamma_2 L^2 - \gamma_1 L)$. However, when the SDEM3 noise model is considered, both the twisted and untwisted code have only an exponential decay. We note that γ_1 and γ_2 become larger for smaller



Supplementary Figure 14. **Logical failure probability p_L of the X^3Z^3 Floquet code with twisted periodic boundary conditions as a function of $x = (p - p_{th})L^{1/\nu}$.** Data shown are for different lattice lengths L (lattice sizes $L \times 3L/2$). The plots are calculated for two different noise models: code capacity (upper panels) and SDEM3 (lower panels), with various bias strengths: (a,e) $\eta = 1$, (b,f) $\eta = 9$, (c,g) $\eta = 999$, and (d,h) $\eta = \infty$. We fit the results to the function $A + Bx + Cx^2$ (blue line). Insets: Logical failure probability p_L vs physical error rate p ; thresholds p_{th} are shown by magenta vertical lines. Results for each lattice length L and physical error rate p are averaged over $10^6 - 10^7$ number of shots.



Supplementary Figure 15. **Sub-threshold logical failure probabilities p_L vs lattice length L for X^3Z^3 Floquet codes with different periodic boundary conditions.** Data for untwisted and twisted code are shown in blue and magenta, respectively. Results are calculated for two different noise models: (a) code-capacity and (b) SDEM3 noise at infinite bias $\eta = \infty$. They are also computed with various physical error rates p . All curves can be fitted to decay functions $f \propto \exp(-\gamma_2 L^2 - \gamma_1 L)$ or $f \propto \exp(-\gamma_1 L)$ where $\gamma_{1,2}$ are increasing functions of $(p_{th} - p)$. Each data point is averaged over $10^5 - 10^{10}$ number of shots.

subthreshold physical error rates.

The super-exponential and exponential decay dependences of the twisted X^3Z^3 code's logical failure probability with respect to the lattice length L under the code-capacity and SDEM3 noise models, respectively, are due to the fact that the code distances scale quadratically and linearly with the lattice length under these two different

noise models. From the “shortest graph-like error” generated via Stim [11], we find that the measurement and Pauli errors in the SDEM3 model can conspire to produce logical error strings with lengths that scale linearly with the lattice length. Using Stim, we also calculate the size of the shortest graph-like error (i.e., the code distance) of the twisted X^3Z^3 code under the infinitely Z -biased

noise and find the following relations:

$$d_{\text{twist}} = \frac{d_{\text{untwist}}^2}{2} = \frac{L^2}{2}, \quad (6a)$$

$$d_{\text{eff,twist}} = 2d_{\text{eff,untwist}} - 1 = L - 1, \quad (6b)$$

for the code-capacity and SDEM3 noise models, respectively. Here d_{twist} and $d_{\text{eff,twist}}$ are the code distances for the twisted X^3Z^3 code under the code-capacity and SDEM3 noise, respectively. Similarly, d_{untwist} and $d_{\text{eff,untwist}}$ are the code distances defined for the untwisted code, while L is the length of the honeycomb lattice with size $L \times 3L/2$. We note that for $\eta < \infty$ of

both noise models, the size of the shortest graph-like error calculated from Stim is the same for both the untwisted and twisted codes.

In addition to the subthreshold scaling, as can be seen from Supplementary Fig. 15, for a fixed physical error rate under the infinitely phase-biased code-capacity noise, the logical failure probabilities of the twisted code are much lower than those of the untwisted case while they are roughly similar for the case of SDEM3 noise. The reason is because the infinite-bias code-capacity threshold for the twisted code is higher than that of the twisted code while they are the same for the SDEM3 case, as shown in Supplementary Fig. 14.

-
- [1] C. Gidney, M. Newman, A. Fowler, and M. Broughton, A fault-tolerant honeycomb memory, *Quantum* **5**, 605 (2021).
- [2] M. B. Hastings and J. Haah, Dynamically generated logical qubits, *Quantum* **5**, 564 (2021).
- [3] M. Davydova, N. Tantivasadakarn, and S. Balasubramanian, Floquet codes without parent subsystem codes, *PRX Quantum* **4**, 020341 (2023).
- [4] M. S. Kesselring, J. C. Magdalena de la Fuente, F. Thomsen, J. Eisert, S. D. Bartlett, and B. J. Brown, Anyon condensation and the color code, *PRX Quantum* **5**, 010342 (2024).
- [5] S. Puri, L. St-Jean, J. A. Gross, A. Grimm, N. E. Frattini, P. S. Iyer, A. Krishna, S. Touzard, L. Jiang, A. Blais, *et al.*, Bias-preserving gates with stabilized cat qubits, *Science advances* **6**, eaay5901 (2020).
- [6] K. Sahay, J. Jin, J. Claes, J. D. Thompson, and S. Puri, High-threshold codes for neutral-atom qubits with biased erasure errors, *Phys. Rev. X* **13**, 041013 (2023).
- [7] A. Fahimniya, H. Dehghani, K. Bharti, S. Mathew, A. J. Kollár, A. V. Gorshkov, and M. J. Gullans, Fault-tolerant hyperbolic Floquet quantum error correcting codes, [arXiv:2309.10033](https://arxiv.org/abs/2309.10033) (2023).
- [8] X. Fu and D. Gottesman, Error correction in dynamical codes, [arXiv:2403.04163](https://arxiv.org/abs/2403.04163) (2024).
- [9] J. P. Bonilla Ataides, D. K. Tuckett, S. D. Bartlett, S. T. Flammia, and B. J. Brown, The XZZX surface code, *Nature Communications* **12**, 2172 (2021).
- [10] D. Forlivesi, L. Valentini, and M. Chiani, Logical error rates of XZZX and rotated quantum surface codes, *IEEE Journal on Selected Areas in Communications* **42**, 1808 (2024).
- [11] C. Gidney, Stim: a fast stabilizer circuit simulator, *Quantum* **5**, 497 (2021).