Fragile boundaries of tailored surface codes

Oscar Higgott,[1,2,*] Thomas C. Bohdanowicz,[3,4] Aleksander Kubica,[4,5]
Steven T. Flammia,[4,5] and Earl T. Campbell[6,7,8]

[1] *University College London*
[2] *AWS Center for Quantum Computing, Cambridge, UK*
[3] *Goldman, Sachs & Co., New York, NY, USA*
[4] *AWS Center for Quantum Computing, Pasadena, CA, USA*
[5] *California Institute of Techonology, Pasadena, CA, USA*
[6] *AWS Center for Quantum Computing, Cambridge CB1 2GA, United Kingdom*
[7] *Riverlane, Cambridge, United Kingdom*
[8] *Department of Physics and Astronomy, University of Sheffield, Sheffield S3 7RH, United Kingdom*

(Dated: March 10, 2022)

Biased noise is common in physical qubits, and tailoring a quantum code to the bias by locally modifying stabilizers or changing boundary conditions has been shown to greatly increase error correction thresholds. In this work, we explore the challenges of using a specific tailored code, the XY surface code, for fault-tolerant quantum computation. We introduce an efficient and fault-tolerant decoder, belief-matching, which we show has good performance for biased circuit-level noise. Using this decoder, we find that for moderately biased noise, the XY surface code has a higher threshold and lower overhead than the square CSS surface code, however it performs worse when below threshold than the rectangular CSS surface code. We identify a contributor to the reduced performance that we call fragile boundary errors. These are string-like errors that can occur along spatial or temporal boundaries in planar architectures or during logical state preparation and measurement. While we make partial progress towards mitigating these errors by deforming the boundaries of the XY surface code, our work suggests that fragility could remain a significant obstacle, even for other tailored codes. We expect belief-matching will have other uses, and find that it increases the threshold of the surface code to 0.940(3)% in the presence of circuit-level depolarising noise, compared to 0.817(5)% for a minimum-weight perfect matching decoder.

I. INTRODUCTION

Quantum error correcting codes should be tailored to exploit the structure in the noise present in the physical systems used to realise them [1–4]. Several methods have been proposed for exploiting this bias, with the aim of increasing thresholds or reducing the qubit overhead below threshold [5–10]. One approach to tailoring codes for biased noise is to modify the basis of the stabilizer measurements in the surface code, while retaining the same square lattice layout. The XY surface code and XZZX surface codes both follow this approach, and have been shown to have extremely high thresholds under biased noise [5–9]. Another approach is to apply schedule-induced gauge fixing to subsystem codes, which was shown to achieve high thresholds for the subsystem surface code in Ref. [10] with biased noise.

In this work, we tackle the problem of choosing the optimal variant of the surface code in the presence of biased noise when qubits are constrained to a square lattice geometry and open boundary conditions. Inspired by the biased noise present in some quantum devices [4], we consider a circuit-level noise model containing two parameters: a noise strength p and a bias η. The bias η is the quotient of the probability that some Z-type error occurs, and the probability that any other occurs (for $P \in \{X, Y, Z\}$, a P-type Pauli operator on n qubits is an operator in the set $\{I, P\}^{\otimes n}$). We are primarily interested in an optimization with respect to the required qubit overhead below threshold, in parameter regimes where useful fault-tolerant quantum computation is feasible. However, we do also compare the thresholds of the codes considered.

The variants of the surface code we study are the standard Calderbank-Shor-Steane (CSS) surface code, as well as the XY surface code, which uses Y-type stabilizers in place of Z-type stabilizers [5–7], both shown in Figure 1. For the CSS surface code, we allow the aspect ratio of the lattice to be optimized to reduce the qubit overhead below threshold. For example, the X distance can be reduced relative to the Z distance for Z-biased noise (where for $P \in \{X, Y, Z\}$, the P distance of a code is the minimum weight of a non-trivial P-type logical operator). When the aspect ratio of a surface code is optimized in this way, we will refer to it as a *rectangular* surface code. For the XY surface code we consider only a square lattice geometry, since the aspect ratio here determines the X and Y distance, which should be equal for our chosen noise model in which X and Y errors are equiprobable.

The XY surface code has so far only been studied in an idealised setting of perfect syndrome measurements or a phenomenological noise model [6, 7]. To assess the practicality of the XY surface code for fault-tolerant quantum computing, it is important to develop a decoder for it that is tailored to biased *circuit-level* noise. Furthermore, it is crucial to study how logical operations, such

* oscar.higgott.18@ucl.ac.uk

as logical state preparation, measurement, lattice surgery and magic state distillation can be implemented with the XY surface code, in a way that takes advantage of the noise bias. Additionally, we are interested not just in the threshold of the code, but also in the qubit overhead required to achieve a desired logical error rate, and how this compares to alternative approaches.

Here we show how belief-propagation (BP) can be combined with minimum-weight perfect matching MWPM to decode biased circuit-level noise in the XY surface code. Our decoder applies belief propagation to a Tanner graph describing the circuit-level noise. Whenever BP fails to converge, we use the marginal probabilities output by BP to determine the edge weights in the matching graph for a MWPM decoder. Since our decoder combines belief-propagation and matching decoders, we refer to it as the belief-matching decoder. Belief-matching has conceptual similarities to the decoder proposed by Criger and Ashraf [11], which considered an idealised noise model with perfect syndrome measurements. Here we extend this idea and show that more realistic circuit level noise can also be handled by belief-matching. We find that belief-matching significantly outperforms MWPM alone, and we observe a threshold of 0.841(6)% CNOT infidelity for biased circuit-level noise with bias $\eta = 100$. This constitutes a 1.69× relative improvement on the 0.498(1)% threshold observed using MWPM only for the CSS surface code under the same noise model. Our decoder also improves on MWPM for depolarising circuit-level noise in the CSS surface code, achieving a threshold of 0.940(3)%, compared to 0.817(5)% for MWPM alone. We also compare the performance of belief-matching to that of a maximum-likelihood (ML) decoder tailored to circuit-level noise on small code sizes for benchmarking purposes.

One reason the XY surface code is so promising for biased noise is that the Z distance of the code is equal to the number of data qubits n, improving on the $O(\sqrt{n})$ Z distance scaling of the square CSS surface code. Furthermore, it was shown that under pure Z noise, the code is equivalent to the repetition code, and therefore has a threshold of 50% [6]. However, we show that the high tolerance of the XY surface code to Z errors is extremely fragile. This fragility occurs wherever the space-time picture of the XY surface code has a boundary. With open boundary conditions, the XY surface code has a spatial boundary on which we find failure mechanisms that require only $O(\sqrt{n})$ Z errors and a single X or Y error. We refer to these as fragile spatial boundary errors. Using belief-matching, we present numerical results consistent with the conclusion that these failure mechanisms dominate at lower error rates and finite bias. Irrespective of lattice boundary conditions, the XY surface code will always have a temporal boundaries that be an interpreted as state preparation and measurement (SPAM) operations. We find string-like Z errors that can occur on these temporal boundaries even at infinite bias, during which either the X or Y type stabilizers cannot be measured. This leads to an effective $O(\sqrt{n})$ Z distance during SPAM operations. Again, our belief-matching decoder reveals that these SPAM errors dominate over memory errors for a logical idle of d rounds of stabilizer measurement. Temporal boundaries also arise during lattice surgery [12–15] and so these operations are also vulnerable. We refer to this family of errors as fragile temporal boundary errors. Our work establishes that boundaries are fundamentally fragile in the XY surface code, and more generally one can construct problematic fault patterns that traverse both temporal and spatial boundaries of the code. None of the prior art reviewed above [5, 7] considered the below-threshold error scaling at finite-bias with open boundary conditions or the error scaling of logical SPAM errors. Consequently these previous works do not observe any of the dominant error mechanisms reported here.

We compare the qubit overhead of the XY surface code to the square and rectangular CSS surface codes below threshold for a bias of $\eta = 100$. We find that the XY surface code outperforms the square CSS surface code in terms of qubit overhead for all physical error rates. However, for CNOT infidelities below around 0.4%, we find that the rectangular CSS surface code outperforms the XY surface code, owing to the reduction in qubit overhead achieved by optimizing the aspect ratio of the lattice.

II. DECODING CIRCUIT-LEVEL NOISE

For the XY surface code, a single qubit Z error anticommutes with four stabilizers in the bulk, rather than two as in the CSS surface code. This makes decoding more challenging, since minimum-weight perfect matching or union-find [16] cannot be applied (without modification) in a way that takes advantage of the bias. In Ref. [5], the tensor network decoder of Bravyi *et al.* [17] (the BSV decoder) was used to decode the XY surface code for finite (and infinite) bias, with the threshold shown to approach the hashing bound for all values of bias. However, the BSV decoder does not handle imperfect stabilizer measurements and has a high computational complexity, limiting its usefulness for practical fault-tolerant quantum computing. Tuckett *et al.* developed a decoder for the XY surface code that handles noisy stabilizer measurements and has improved thresholds at finite and infinite bias, relative to the MWPM decoder [7]. Their decoder uses symmetries present in the syndrome at high bias to decompose the decoding problem into MWPM subroutines that exploit the bias to improve decoding performance. However, although the performance of the decoder is promising for decoding phenomenological noise, it is not clear how well it can generalise to handle circuit-level errors in stabilizer measurement circuits.

In this section, we first review the BP algorithm and then discuss how it is combined with minimum-weight

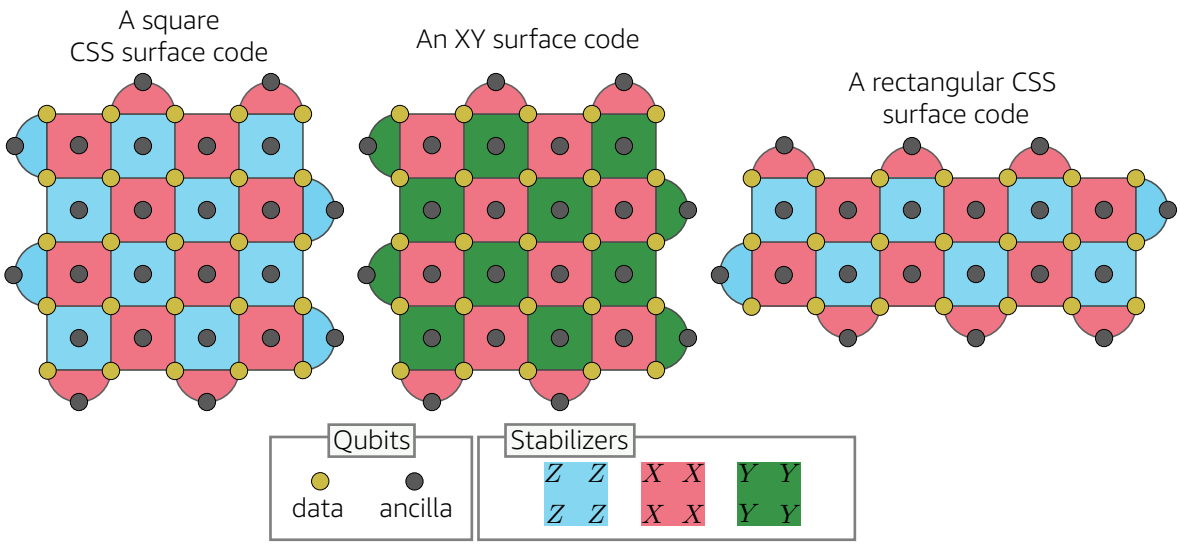


FIG. 1. The three code families compared in this work. The square, CSS surface code is the most commonly encountered surface code, without any tuning for the noise bias. We denote the lattice size by L, and here the square CSS and XY surface codes both have $L = 5$.

perfect matching to decode circuit-level noise.

A. BP review

The BP algorithm, also known as the *sum-product algorithm*, is an efficient iterative message passing algorithm with good performance for decoding classical low-density parity check (LDPC) codes [18]. Consider a binary check matrix H defining a linear code $\ker(H)$. BP is most readily understood by considering the *Tanner graph* $\mathcal{T}(H)$ of the check matrix H. The Tanner graph is a bipartite graph with a *check node* for each parity check (row of H), and a *variable node* for each bit (column of H), and graphically represents a factorisation of the joint probability distribution over the bits. Each check node is connected by an edge to the variable nodes corresponding to the bits it acts nontrivially on. The BP algorithm takes as input the prior probabilities that each bit is flipped, as well as the syndrome of each parity check. Each iteration of BP consists of a *horizontal* step and a *vertical* step. In the horizontal step, each check node (a row of H) sends a message to its adjacent variable nodes. In the vertical step, each variable node (a column of H) sends a message to its adjacent check nodes, where each message is essentially a local application of Bayes' rule. In each iteration, the latest check-to-variable messages can be used, along with the priors, to compute the 'pseudoposterior probabilities', which approximate the marginal probabilities that each bit has been flipped, given the priors and the syndrome.

To improve the numerical stability and efficiency of BP, we use log-likelihood ratios (LLR) to represent probabilities and compute messages, where the LLR of a binary random variable U is defined as

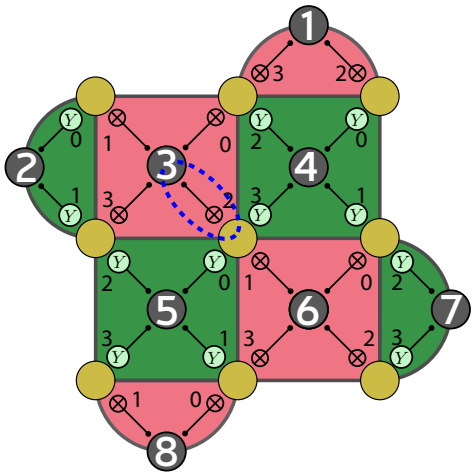
$$L(U) = \log\left[\Pr(U = 0)/\Pr(U = 1)\right]. \tag{1}$$

We denote by q_i the LLR of the pseudoposterior probability that bit i was flipped and define a binary vector $\mathbf{x}$ of *hard decisions* where element $\mathbf{x}[i]$ is set to 0 if $q_i > 0$ and is set to 1 if $q_i \leq 0$. In each iteration of BP we compute $H\mathbf{x}$, and stop the algorithm and return $\mathbf{x}$ if $H\mathbf{x} = \mathbf{s}$, where $\mathbf{s}$ is the syndrome. When this happens we say that BP has *converged*. If a maximum number of iterations m_{iter} is reached without BP converging, then we record a heralded failure (and we set $m_{\text{iter}} = 30$ in this work). We refer the reader to Refs. [19, 20] for a more detailed overview of BP and its variants.

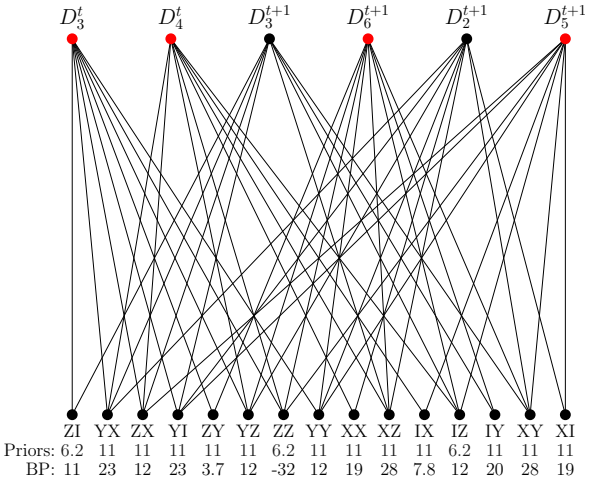
While BP is an effective decoder for classical LDPC codes, its application to quantum codes faces challenges. Most notably, the marginals output by BP cannot be used to distinguish between multiple equiprobable solutions to the decoding problem that differ by stabilizers [21]. Several modifications of BP have been used with the purpose of fixing the problem that quantum degeneracy poses for the BP decoder, most notably the use of *ordered statistics decoding* (OSD) post-processing of the BP posterior marginal probabilities [22], which was successfully used to decode hypergraph product codes in Refs. [23–25].

B. Belief-matching

In this work, we apply BP to a Tanner graph describing the biased circuit-level noise model of the XY surface code using the posterior probabilities output by BP to choose edge weights for a MWPM decoder. Belief-propagation has been combined with MWPM before in



(a)



(b)

FIG. 2. (a) A layout and schedule for measuring the stabilizers of a $L = 3$ XY surface code. The stabilizers are numerically labelled 1-8 at the corresponding ancilla vertex. We show the control-not and control-Y gates used to measure these stabilisers with numerical labels 0-3 indicating the time ordering of these gates. We define a detector D_j^t as the parity of stabilizer j in consecutive rounds $t-1$ and t. (b) The circuit-level Tanner graph corresponding to the circuit in (a). We show only a small subgraph of the full Tanner graph, corresponding to the two-qubit Pauli errors that can occur after the highlighted CNOT gate in round t in (a). Below each variable node, we also show the log-likelihood ratio (LLR) of its prior, as well as the LLR of the posterior probability estimate output by BP given the syndrome in which the red check nodes are flipped. The BP hard decisions here would output ZZ as a correction.

Ref. [11], where belief-propagation posteriors were used for multi-path summation to produce edge weights for a MWPM decoder, finding a threshold of 17.76% for the surface code with depolarising noise and perfect syndrome measurements. However, Ref. [11] did not consider how to generalise the method to handle noisy gates

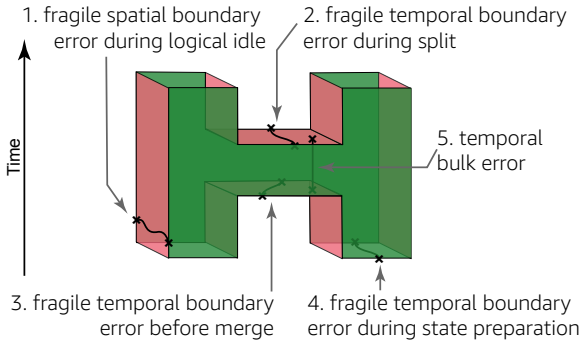


FIG. 3. A space time diagram for two XY surface codes patches being prepared and then undergoing lattice surgery merging and splitting. We illustrate 5 different string-like errors that can lead to logical faults. Errors 1–4 are all purely boundary effects, constrained to either spatial (vertical) or temporal (horizontal) boundaries, and contain $O(\sqrt{n})$ Z errors and at most one X or Y error. Error 5 is a sequence of measurement errors and we observe that these can also form strings between pairs of time-boundaries using τ measurement failures, where τ is the number of stabilizer rounds used during lattice surgery.

in the syndrome extraction circuit.

In the circuit-level Tanner graph, each *check node* is defined to be a linear combination of measurement outcomes in a stabilizer measurement circuit, which we will refer to as a *detector* [26]; a detector is also referred to as an *error-sensitive event* in the literature [27]. For example, when decoding the surface code in the presence of noisy syndrome measurements (such as circuit-level noise), the linear combination of each pair of consecutive stabilizer ancilla measurements is taken to define a detector. A detector could also be the parity of gauge operator measurements that factorise a stabilizer in a subsystem code. We say that a detector has *flipped* if its binary value differs from the value it would take in an error-free stabilizer measurement circuit. Each *variable node* in the circuit-level Tanner graph corresponds to a different type of fault that can occur in the stabilizer measurement circuit, and is adjacent to the set of check nodes corresponding to the detectors that the fault flips. If multiple fault patterns trigger the same set of detectors (and are thus indistinguishable), then these fault patterns are merged into a single variable node which is assigned a probability equal to the probability that an odd number of the faults occurred. A circuit-level Tanner graph is a graphical representation of a *detector error model* in Stim [26], but where equivalent variable nodes have been merged as just described.

Figure 2 shows a Tanner graph for a biased circuit-level noise model, restricted to the 15 non-trivial two-qubit Pauli errors that can occur after a single CNOT gate in the parity check measurement schedule (it is a small subgraph of the full circuit-level Tanner graph). A Tanner graph describes a factorisation of a joint probability distribution in which each bit (corresponding to a

variable node) is flipped *independently* with the assigned prior probability. Note that, in the standard Pauli circuit noise models considered in the literature and in this work, the probabilities of each Pauli error that can occur after a gate are described as probabilities of disjoint errors, rather than as independent events. While some specific Pauli noise models, such as the depolarising noise model, *can* be described as an independent distribution [28], this is not the case in general. When computing priors and constructing the Tanner graph, we make the approximation that each probability of a disjoint error mechanism instead corresponds to the probability of an independent error mechanism. This approximation is correct to leading order in p, and therefore a good approximation for the physical error rates we considered.

Due to low weight degenerate errors, BP on its own does not have a threshold for the surface code, and we confirmed that this is also the case for the XY surface code with circuit-level noise. We recover a threshold (see Section IV) by using the BP posteriors to choose edge weights for MWPM, which we only run on instances where BP fails to converge (almost all failures for BP alone are these heralded failures). Each edge in the matching graph used by MWPM corresponds to a variable node of degree one or two in the Tanner graph (a degree one node corresponds to an edge connected to a boundary). The circuit-level Tanner graph for the XY (and CSS) surface code also contains variable nodes with degree greater than two, which would correspond to hyperedges in a decoder hypergraph (the obvious generalisation of a matching graph). However, these hyperedges can always be approximated by a sets of edges already present in the matching graph [26]. For example, consider a weight-four hyperedge (t, u, v, w), and assume that the edges (t, u) and (v, w) are already present in the matching graph, we say that the hyperedge (t, u, v, w) can be decomposed into the edges (t, u) and (v, w). To construct the edge weights in the matching graph, we first assign the BP posteriors of degree-two and degree-one variable nodes in the Tanner graph to the corresponding edges in the matching graph. Then, for each hyperedge, we add its BP posterior to the probabilities already assigned to each of the edges in its decomposition (and set the probability to one if it exceeds one). Finally, we assign each edge a weight $-\log(p)$, where p is the probability assigned to the edge. Since the advantage that belief-matching offers over MWPM derives from its use of *hyperedges* present in the circuit-level Tanner graph, we expect its advantages over MWPM to be most prominent for error models where these hyperedge error mechanisms occur with probabilities comparable to or greater than the probabilities of edge-like error mechanisms. Therefore, belief-matching is well suited to decode the XY surface code under Z bias (or the CSS surface code with depolarising noise), but not the CSS surface code with Z biased noise (for which dominant error mechanisms are all edge-like).

The asymptotic running time of belief-matching is dominated by the MWPM subroutine, since the running time of BP is linear in the number edges in the circuit-level Tanner graph. The BP step, although linear time, can still be quite computationally intensive, since the number of edges in the circuit-level Tanner graph is a constant factor larger than the number of edges in the corresponding matching graph, and running time does not depend strongly on the weight of the syndrome (it is not necessarily faster at low p, unlike MWPM). However, we do not expect this to be an issue since BP is highly parallelisable, and very fast implementations are widely used for decoding classical LDPC codes.

C. ML decoding

We benchmark the performance of belief-matching against a maximum likelihood (ML) decoder for circuit-level noise, which outputs a Pauli correction that maximises the probability that the combined error and correction is in the stabilizer group. We give the ML decoder the problem of decoding $L-1$ rounds of noisy stabilizer measurements, followed by a round of perfect stabilizer measurements. After obtaining a set of noisy syndromes from the first $L-1$ rounds, an n-qubit Pauli error E has accumulated on the code block from the execution of the measurement circuits. The final perfect round of syndrome measurement extracts the true syndrome of the error E. Let T be an n-qubit Pauli operator consistent with the true syndrome. A circuit-level ML decoder finds a logical operator $\overline{L} \in \mathcal{C}(\mathcal{S}) \setminus \mathcal{S}$ that maximises $\Pr([T\overline{L}])$, returning $T\overline{L}$ as the correction. Here, the centralizer $\mathcal{C}(\mathcal{S})$ is the set of n-qubit Pauli operators that commute with all elements of $\mathcal{S}$ and the probability $\Pr([P])$ of the equivalence class $[P] \coloneqq \{PS : S \in \mathcal{S}\}$ is defined as $\Pr([P]) \coloneqq \sum_{S\in\mathcal{S}} \Pr(PS)$, where $\Pr(P)$ is the probability that the error P has accumulated on the code block, given the full syndrome and knowledge of the circuit-level noise model.

The ML decoder we implemented is the tensor network decoder described in Ref. [29], which can be seen as a generalization of the BSV decoder of Bravyi *et al.* [17] to the setting of imperfect syndrome measurements and circuit-level noise. The decoder is constructed by modelling all of the individual fault locations of the stabilizer measurement circuit with individual tensors whose entries are probabilities of different Pauli errors having occurred, as defined in the circuit-level error model. By using the mathematical structure of a subsystem code called the *circuit history code* [30], which is determined by our stabilizer measurement circuit, these individual fault tensors can be interconnected to a set of Kronecker delta tensors resulting in a tensor network which upon contraction allows us to find out the solution to the maximum-likelihood decoding problem. Exact tensor network contraction, as with any approach to exact ML decoding, is computationally expensive. But the tensor network approach to ML decoding offers the advantage of being

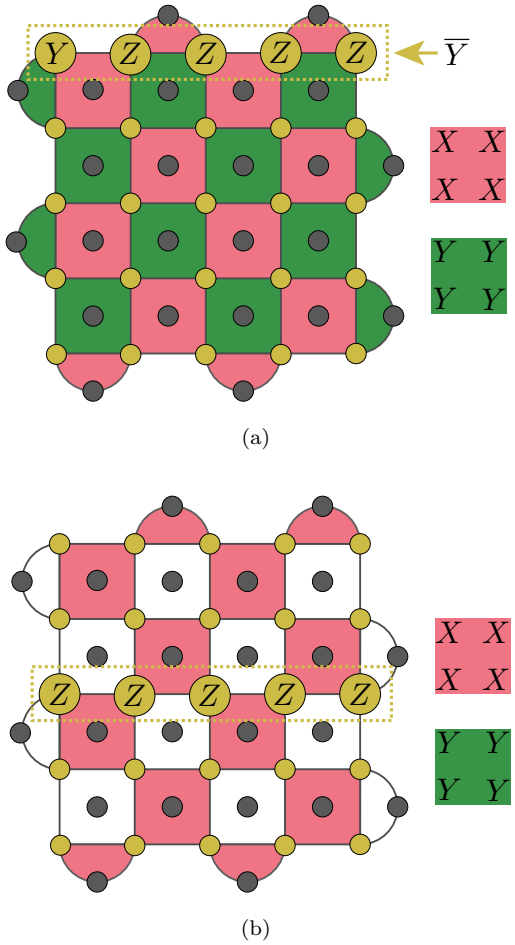


FIG. 4. Two types of fragile boundary errors. (a) A fragile spatial boundary error that can occur at finite bias, involving a single Y error, and $O(\sqrt{n})$ Z errors. (b) A fragile temporal boundary error. It is a Z-type logical error with weight $O(\sqrt{n})$ that can occur during a logical X measurement, when only X-type stabilizers are being measured.

able to use numerical methods for approximate tensor network contraction to lower the complexity of the calculation while maintaining a high degree of accuracy, which can be controlled.

Before presenting our numerical results, we will now discuss some low weight error mechanisms that can occur in the XY surface code, even at high bias.

III. FRAGILITY OF THE XY SURFACE CODE

In this section, we will show that the protection provided by the XY surface code is *fragile*, meaning that there are failure mechanisms in the XY surface code that require only $O(\sqrt{n})$ Z errors during logical state preparation and measurement, as well as during logical idling at finite bias. An overview of all the errors discussed in this section is presented in Figure 3.

A. Finite bias and spatial fragile boundary errors

At finite bias, errors that consist of a mix of X, Y and Z Pauli operators are common failure mechanism. In this section, we give examples of logical operators consisting of a single X or Y error and $O(\sqrt{n})$ Z errors. An example of a logical Y error of this form is shown in Figure 4a. A similar logical Y error can occur on the south boundary, and likewise logical X errors consisting of a single X and $O(\sqrt{n})$ Z operators can occur on the east and west boundaries. We will refer to any of these error patterns as *fragile spatial boundary* errors as they only occur at planar code spatial boundaries and highlight the fragility of the infinite bias limit. In Figure 3, error 1 is such an error. At low physical error rates and high bias, we would expect fragile spatial boundary errors to be dominant failure mechanisms. Furthermore, since these failure mechanisms occur on all four boundaries, we expect a square aspect ratio to be optimal (assuming X and Y error probabilities are similar). The existence of fragile spatial boundary errors emerges from the open boundary conditions.

B. Temporal boundary errors and logical state preparation and measurement

In order to measure the logical X operator fault-tolerantly in the XY surface code, we measure all data qubits in the X basis and infer both the X logical operator and X stabilizers in post-processing. Similarly, we measure the logical Y operator fault-tolerantly by measuring all data qubits in the Y basis.

Consider the measurement of a logical X operator (an analogous argument applies to logical Y operator measurements). Since the X stabilizers are inferred from classical post-processing of data qubit measurements, rather than using an ancilla and measurement circuit, the X stabilizers can be measured perfectly in this final round (and data qubit measurement errors can be interpreted as data qubit memory errors). However, we cannot infer anything about the Y stabilizers in this final round, since we measured the data qubits in the X basis. Since we measure only half of all the stabilizers, we no longer retain an $O(n)$ Z distance at infinite bias. In Figure 3, error 4 is such an error. In Figure 4b, we show an example of an undetectable $O(\sqrt{n})$ Z-type logical failure mechanism that can occur just before (or during) a logical X measurement, and which flips the outcome of the logical X measurement. The same type of fault can also occur during logical state preparation (e.g. when preparing a logical X eigenstate, data qubits are initialised in $|+\rangle$ states, and so only X stabilizers can be measured initially).

C. Fragility of lattice surgery

Temporal boundaries arise not only during logical state preparation and measurement, but also during lattice surgery operations. Lattice surgery is a computational primitive enabling surface code computation in a 2D layout [12–15] through fault-tolerant measurements of logical multi-qubit Pauli operators. Figure 5 shows two XY surface code patches being merged into a single patch, which is the first step of lattice surgery for measuring a $\overline{Y} \otimes \overline{Y}$ logical observable. Figure 5 highlights a $O(\sqrt{n})$ Z error occurring just before the merge that would go undetected and cause an logical fault. The preparation of physical qubits in the $|+\rangle$ state in this time slice correspond to temporal boundaries in the space-time picture of Figure 3 where error 3 represents a similar temporal boundary error. This is essentially the same fault mechanism as afflicts logical state preparation, which can be seen from comparing errors 3 and 4 in Figure 3.

Most fragile errors encountered have been constrained to boundaries, either temporal or spatial. However, during lattice surgery a logical failure can also occur due to string-like errors propagating through the bulk as illustrated by error 5 of Figure 3. Since these errors terminate at temporal boundaries but travel through the bulk, we refer to them as temporal bulk errors. Note that a vertical error in the space-time picture corresponds to a measurement failure of a stabilizer measurement that occurs with some probability p_m. If we repeat these stabilizer measurements d_m times during lattice surgery, then error 5 of Figure 3 represents d_m consecutive measurement faults and occurs with probability $O(p_m^{d_m/2})$. This fault results in the lattice surgery operation giving an incorrect value of the measured logical multi-qubit Pauli operator.

A standard choice is to set $d_m = \sqrt{n}$, and if p_m is similar to the probability of a Z error, then the probability of each such temporal bulk errors is comparable to a $O(\sqrt{n})$ Z error. However, there are more possible temporal bulk errors since there are more paths through the bulk than along the boundaries. Of course, temporal bulk errors can be suppressed by having more rounds of stabilizer measurements during lattice surgery (e.g. setting $d_m = n$) but this results in significantly slower quantum computation.

D. Performance below threshold

We expect the fragile boundary errors of Section III A to have a significant impact on the performance of the code below threshold. For simplicity, consider a noise model where the probability p of a single-qubit Z error is low, nevertheless it is substantially higher than the probability p/η of a single-qubit X or Y error. The temporal boundary errors described in Section III B are equivalent to the dominant failure mechanisms in the square CSS surface code, and decay as $O(p^{\sqrt{n}/2})$. We expect fragile

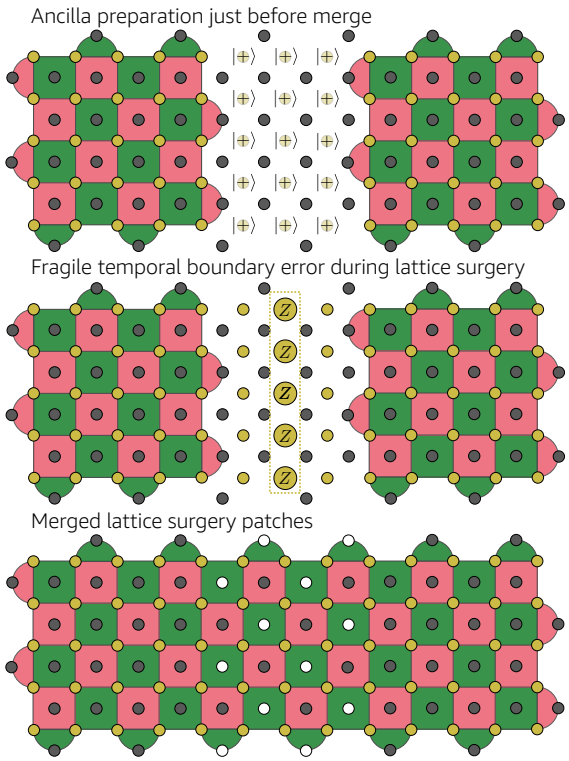


FIG. 5. We illustrate the merge step of lattice surgery performing a logical $\overline{Y} \otimes \overline{Y}$ measurements between two square XY surface code patches, including a possible fragile temporal boundary error. Before the merge, the data qubits between patches must be prepared in the $|+\rangle$ state. We illustrate a possible fragile temporal boundary error that occurs during or after the $|+\rangle$ state preparation but before the merge stabilizers are measured. During the merge step, the vertices highlighted white have random outcomes except that their product gives the outcome of the logical $\overline{Y} \otimes \overline{Y}$ measurement.

spatial boundary errors to decay as $O\left(p^{\sqrt{n}/2+O(1)}/\sqrt{\eta}\right)$ with minimum-weight decoding far below threshold. To understand why this is the case, consider the most likely logical operator E_d spanning the lattice that comprises $O(\sqrt{n})$ Z errors and one X or Y errors. We can split E_d into two operators E_a and E_b, where E_a comprises one X or Y error and c Z errors and E_b comprises $\sqrt{n}-1-c$ Z errors, where $c \in [0\,..\,\sqrt{n}-1]$. We choose c such that E_a and E_b both occur with probability $O\left(p^{\sqrt{n}/2+O(1)}/\sqrt{\eta}\right)$. Since E_a and E_b cannot be simultaneously correctable, the logical failure will be due to one of them occurring. In most regimes of practical interest, we expect these string-like failure mechanisms to dominate over weight n Z-type logical errors, which decay as $O(p^{n/2})$. In order for weight n Z-type logical errors to dominate we would expect $O(p^{n/2}) \gg O\left(p^{\sqrt{n}/2+O(1)}/\sqrt{\eta}\right)$, which requires an extremely high bias η that seems unachievable in any physical system. As well as considering most-likely errors, it is important also to consider entropic contribu-

tions to the logical error rate as well as the threshold, which are taken into account by our numerical simulations in Section IV. Furthermore, it is important to compare the overhead of the XY surface code to the *rectangular* CSS surface code, for which optimizing the aspect ratio can reduce the qubit overhead.

IV. NUMERICAL SIMULATIONS

In this section, we present numerical simulations that compare the performance of the XY surface code with the square and rectangular CSS surface codes using a biased circuit level noise model. We used Stim to construct the detector error models, decompose hyperedges into edges and sample from the stabilizer measurement circuits [26]. We used PyMatching to decode with MWPM [31].

A. Noise model

We used the same biased circuit-level noise model as in Ref. [15] that is captured by two parameters p and η; there are, however, alternative definitions [32]. Namely, each two-qubit gate is followed by a two-qubit Pauli channel, for which ZZ, ZI or IZ can occur with probability $p/15$ each, and the remaining 12 non-trivial two-qubit Paulis can each occur with probability $\frac{p}{15\eta}$. Each single qubit gate location or single qubit idle location of the same duration (a single time step) is followed by a Z error with probability $p/3$ or an X or Y error each with probability $\frac{p}{3\eta}$. A $|+\rangle$ state is incorrectly prepared as a $|-\rangle$ state with probability $2p/3$, and a $|0\rangle$ state is incorrectly prepared as a $|1\rangle$ state with probability $\frac{2p}{3\eta}$. Each single-qubit X-basis measurement is flipped with probability $2p/3$, and each single-qubit Z-basis measurement is flipped with probability $\frac{2p}{3\eta}$. Each single-qubit gate and two-qubit gate has a duration of a single time step, whereas single-qubit state preparation and measurement are each taken to have a duration of half a time step. For the XY code measurement schedule we used, X stabilizers are measured using CNOT gates controlled on an ancilla initialised in a $|+\rangle$ state, and Y stabilizers are measured using controlled-Y (CY) gates, also controlled on a $|+\rangle$ state. These two-qubit gates are applied in the order indicated by the blue text in Figure 2. For the CSS surface code we used the same schedule for measuring X stabilizers, and Z stabilizers were measured using CNOT gates targeted on an ancilla initialised in the $|0\rangle$ state, and applied in the same order as used for CY gates in the XY surface code schedule. We assume that CY gates can be implemented natively, with the same noise model as CX gates. In general, we have made optimistic assumptions for our XY surface code simulations (perfect logical initialisation and measurement, native CY gates), in order to understand if fragile spatial boundary errors alone result in inferior performance relative to a rectangular CSS surface code (removing these optimistic assumptions will only make performance of the XY surface code worse).

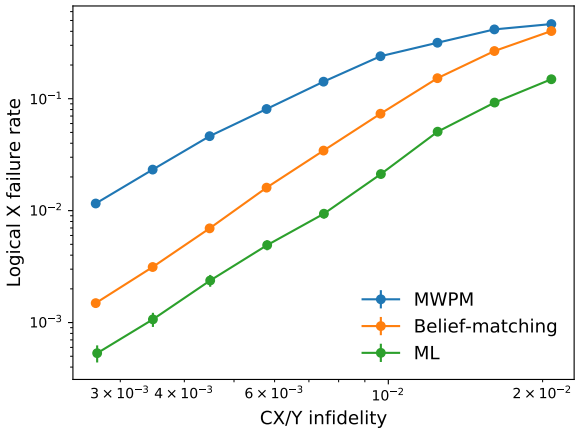


FIG. 6. Performance of the belief-matching decoder compared to a circuit-level ML decoder for a $L = 5$ XY surface code for L rounds with perfect initialisation and syndrome measurements.

B. Comparison with ML decoding

In Figure 6 we compare the performance of the belief-matching decoder with that of pure MWPM and a circuit-level ML decoder, for a $L = 5$ XY surface code for L rounds with perfect initialisation and noisy syndrome measurements. At lower physical error rates (e.g. $p = 0.27\%$), we find that the logical error rate using belief-matching is around 7.8× lower than MWPM alone, and 2.8× higher than ML decoding. The ML decoder was implemented in Julia using PastaQ [33] to approximately contract the tensor network as a matrix product state, fixing a maximum bond dimension of $\chi = 40$ throughout the contraction as we observed no further gains in accuracy by using a larger χ.

C. Thresholds

We compare the threshold using belief-matching for the XY surface code and MWPM for the square, CSS surface code for $\eta = 100$ biased circuit-level noise in Figure 7. We observe a threshold using belief-matching for the XY surface code at 0.841(6)% CNOT infidelity compared to 0.498(2)% for MWPM with the square, CSS surface code, a 1.69× relative improvement. We use the CNOT infidelity p_{CX} when determining and comparing thresholds since it is a useful measure of the noise strength; note that the parameter p only corresponds to the CNOT infidelity for $\eta = 1$, since $p_{\mathrm{CX}} = (\frac{1}{5} + \frac{4}{5\eta})p$. We also see an improvement for depolarising noise ($\eta = 1$) using the square, CSS surface code, with

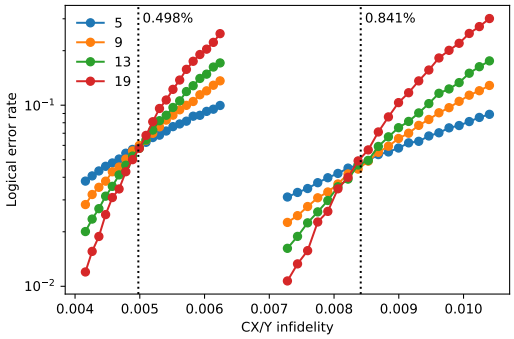


FIG. 7. Threshold of the XY surface code using MWPM (left) and belief-matching (right) for $\eta = 100$.

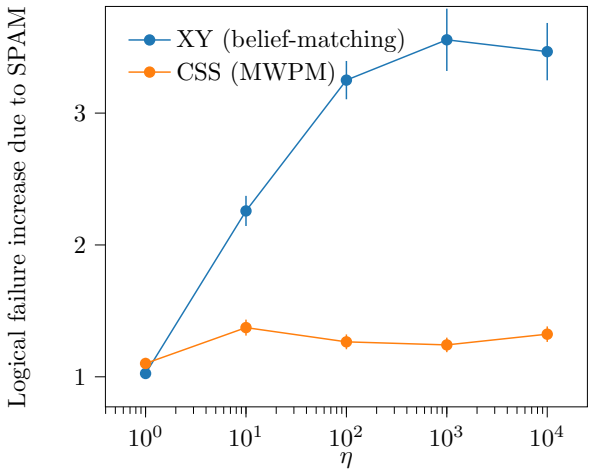


FIG. 8. Effect of SPAM errors on the logical error rate of the $L = 7$ XY and CSS surface codes, for L rounds of noisy syndrome extraction and $p = 0.015$. The y axis shows the ratio $p_{\log}^{\text{SPAM}}/p_{\log}^{\text{mem}}$ where $p_{\log}^{\text{SPAM}}$ is the logical error rate including logical SPAM errors, and $p_{\log}^{\text{mem}}$ is the logical error rate using perfect logical state preparation and measurement.

belief-matching achieving a threshold of 0.940(3)%, compared to 0.817(5)% for MWPM alone. This 1.15× improvement can be attributed to belief-matching taking advantage of correlations between the X and Z matching graphs due to Y errors, and it would be interesting to directly compare the performance of belief-matching to alternative heuristics such as the correlated MWPM decoder introduced in Ref. [34].

D. State preparation and measurement errors

In order to better understand the effect of the $O(\sqrt{n})$ Z-type failure mechanisms present during logical SPAM (fragile temporal boundary errors), we simulated the XY surface code using perfect SPAM, as well as noisy SPAM. For perfect SPAM, we use a round of perfect syndrome extraction at the beginning and end of the computation.

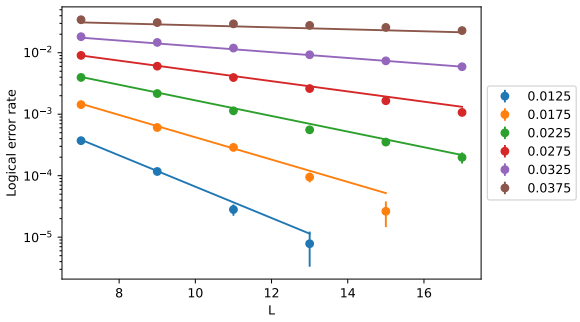


FIG. 9. Logical error rate for a XY surface code with belief-matching.

For noisy SPAM, we initialise data qubits in the $|+\rangle$ state before the first round, and measure data qubits in the X basis at the end of the computation (with physical state preparation and measurement errors occurring at the rate given by the noise model), and all rounds of syndrome extraction circuits are noisy. Using a $L = 7$ XY surface code with L rounds of noisy syndrome extraction, we then calculate the ratio $p_{\log}^{\text{SPAM}}/p_{\log}^{\text{mem}}$, where $p_{\log}^{\text{SPAM}}$ is the logical error rate using noisy SPAM, and $p_{\log}^{\text{mem}}$ is the logical error rate using perfect SPAM. We also carry out the same analysis for the square, CSS surface code decoded using MWPM. As shown in Figure 8, the ratio $p_{\log}^{\text{SPAM}}/p_{\log}^{\text{mem}}$ increases significantly with bias for the XY surface code, but remains small and approximately constant for the square, CSS surface code. This is as expected, since the $Z^{\otimes L}$ errors that can occur during SPAM are more probable than fragile spatial boundary errors.

E. Scaling below threshold

We have also analysed the decay in logical failure rate below threshold, to verify that we observe a decay of the form $O(p^{\sqrt{n}/2}/\sqrt{\eta})$, instead of the $O(p^{n/2})$ scaling we might hope for at infinite bias (without SPAM errors). Our results are shown in Figure 9, where we find that our data is a good fit for an ansatz of the form

$$p_{\log} = a_{\text{tailored}}(b_{\text{tailored}}p)^{(\sqrt{n}+1)/2} \tag{2}$$

for which we find $a_{\text{tailored}} = 0.0419(6)$ and $b_{\text{tailored}} = 24.76(7)$. Here, $p_{\log}$ is the logical Y error rate. Since there is symmetry of the schedule in the bulk (a rotation and reflection of the lattice followed by an exchange of X and Y), we expect (and have numerically verified) the logical X error rate to be almost identical to the logical Y error rate due to the X/Y symmetry of the noise model.

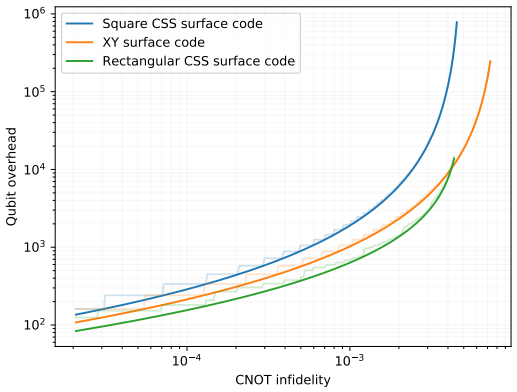


FIG. 10. Qubit overhead of the XY and CSS (square and rectangular) surface codes, as a function of CNOT infidelity, for the biased $\eta = 100$ circuit-level noise model. Translucent, stepped lines permit only odd, integer lattice sizes L, whereas smooth, solid lines interpolate and allow L to be any real positive number. These overhead estimates were computed by solving the fitted ansätze in Equation (2), Equation (3) and Equation (4) for the lattice dimensions, for a target logical error rate of 10^{-12}.

F. Qubit overhead

Using our results for below-threshold scaling of the XY code, and by carrying out a similar analysis for the square and rectangular CSS surface codes, we compare the qubit overhead of the XY surface code with that of the square and rectangular CSS surface codes, to achieve a target logical error rate of 10^{-12} (the "teraquop regime" [35]).

For the rectangular CSS surface code, we fit an ansatz of the form

$$p_{\log}^{X} = \frac{a_x r d_Z}{d_X^2}(b_x p)^{(d_X+1)/2} \tag{3}$$

$$p_{\log}^{Z} = \frac{a_z r d_X}{d_Z^2}(b_z p)^{(d_Z+1)/2} \tag{4}$$

where r is the number of rounds of syndrome extraction, d_X and d_Z are the X and Z distances, and $p_{\log}^{X}$ and $p_{\log}^{Z}$ are the X and Z logical error rates, respectively. For a bias of $\eta = 100$, we found fit parameters $a_x = 0.1015(9)$, $b_x = 42.30(7)$, $a_z = 0.0527(9)$ and $b_z = 1.69(1)$.

We find that the XY surface code outperforms the square CSS surface code in all regimes. However at physical error rates well below threshold, we find that the optimized aspect ratio of the rectangular CSS surface code allows it to outperform the XY surface code. For instance, at a CNOT infidelity of 10^{-3}, the XY surface code requires 1057 physical qubits per logical qubit, a substantial improvement on the 1921 required by the square CSS surface code. However, by using the rectangular CSS surface code (optimizing the aspect ratio), we can achieve even better resource savings, requiring only 681 physical qubits per logical qubit to achieve the same logical error rate.

Therefore, although the XY surface code does still have a higher threshold, and improves over a CSS surface code of the same square aspect ratio, our results suggest that the ability to optimize the aspect ratio can be crucial for taking full advantage of the bias below threshold. Unfortunately, we do not expect to be able to improve the performance of the XY code by changing the aspect ratio, since the failure mechanisms described in Section III can occur both horizontally and vertically.

V. MITIGATING FRAGILE ERRORS

What are the prospects for overcoming these fragility issues with the XY surface code at finite bias? Here, we report partial progress. Recall that the XY surface code is prone to fragile spatial boundary errors composed of a single X or Y error and $\sqrt{n} - 1$ Z errors running along the lattice boundary. We can apply single-qubit Clifford operators along some qubits on the boundary as in Figure 11 so that this boundary error has $\sim \sqrt{n}/2$ Y (or X) errors. More specifically, we apply the Hadamard gate H to one of the two qubits in the support of each Z boundary stabilizer, and the $A \coloneqq HSH$ gate, where S is the phase gate, to one of the two qubits in the support of each X boundary stabilizer. We will refer to this code as the XY surface code with deformed boundaries. Note that it is not important which of the two qubits the Clifford is applied to in each boundary stabilizer, since the two choices are equivalent up to multiplication by the same boundary stabilizer. In Figure 11, we illustrate how a deformed boundary requires more X and Y errors to realise a logical X or Y, and therefore partially mitigates the fragility of spatial boundaries. However, there is a tradeoff. After deforming boundaries, a $\overline{Z}$ can be realised using fewer than n Z errors. As such, boundary deformation will impair performance at infinite bias when only Z errors occur, while providing a performance boost at modest bias. We promised only *partial* progress, since our boundary deformation mitigates fragility of spatial boundaries, but it leaves open whether one can also mitigate against fragile temporal boundaries during SPAM operations and lattice surgery.

To better quantify the effects of boundary deformation, we next consider how the weight of Z-type logical errors scale with the code size. With infinite Z bias, we need only consider the X or Y components of each stabilizer, and whether or not a stabilizer is X-type or Y-type has no impact on its syndrome. We can therefore construct a binary linear code, where the X or Y component of each stabilizer corresponds to the nonzero elements of a parity check, each a row in a check matrix $\mathbf{H}$. If a binary vector $\mathbf{v}$ is in the kernel $\ker(\mathbf{H})$ of $\mathbf{H}$, then the Z-type Pauli operator $\bigotimes_{i=0}^{n-1} Z^{v[i]}$ is either a Z-type stabilizer or Z-type logical operator (here Z-type refers to a Pauli operator in $\{I, Z\}^{\otimes n}$). We can also easily check whether an element in $\ker(\mathbf{H})$ is a stabilizer or non-trivial logical

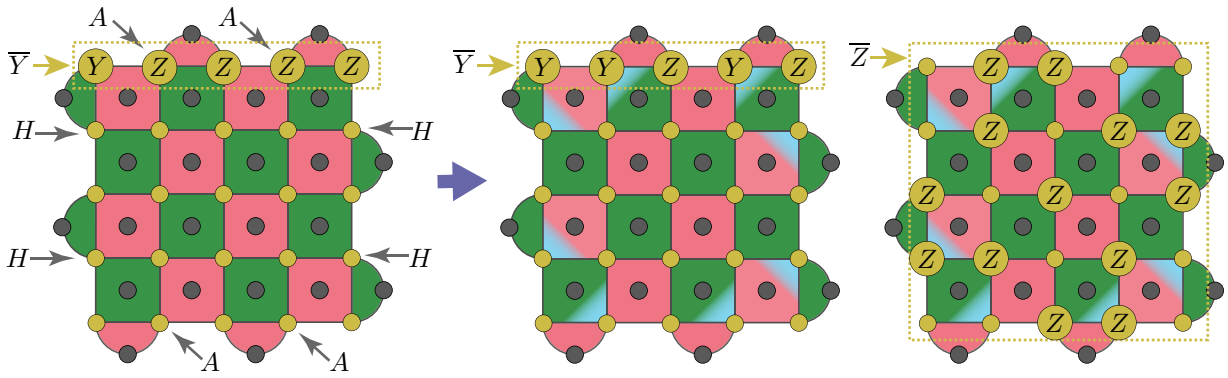


FIG. 11. Deforming the boundary of the XY surface code by applying single-qubit Clifford operators H and A, which swap $X \leftrightarrow Z$ and $Y \leftrightarrow Z$ operators, respectively. On the left, we show the XY surface code and a representative of a logical $\overline{Y}$ prior to the deformation. After the deformation, any logical operator on the boundary comprises at least 3 Pauli X or Y operators, making the XY surface code with deformed boundary more robust. At the same time, a Z-type logical operator can be realised with fewer than n Pauli Z operators, as illustrated in the rightmost example.

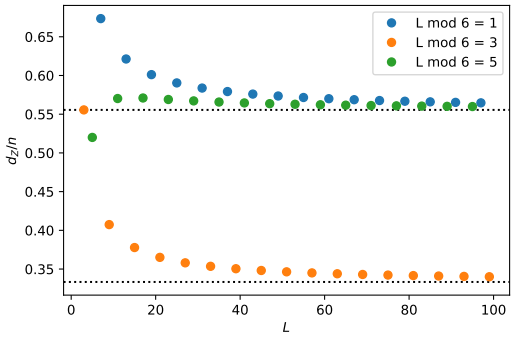


FIG. 12. The Z distance d_Z as a fraction of the number of qubits $n = L^2$ for the XY surface code with deformed boundaries. The horizontal dotted lines are at 5/9 and 1/3.

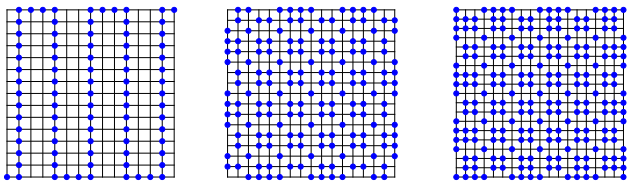


FIG. 13. The Z-type logicals of the XY surface code with deformed boundaries for (a) $L = 15$, (b) $L = 17$ and (c) $L = 19$. Each blue marker denotes a Z operator. Stabilizers are on faces and are of the same form as in Figure 11. For the codes in (b) and (c) the Z-type logical is unique and there are no Z-type stabilizers. For the code in (a) there is also a Z-type stabilizer which, multiplied by the Z logical in (a) gives another Z logical of the same form but rotated 90°.

operator by determining if it commutes with the logical X and Y operators of the code. Using this approach, we computed all Z-type logical operators and stabilizers of the XY surface code with deformed boundaries for all odd $L < 100$. We denote by d_Z the Z distance.

We found that the deformed boundaries degrade the Z distance by only a constant factor, so that it still scales as $\Omega(n)$. In Figure 12 we show the ratio d_Z/n for all odd $L < 100$. For large L, d_Z/n converges to 5/9 if $L \mod 6 = 1, 5$, and converges to 1/3 if $L \mod 6 = 3$. This can be understood by considering the structure of the Z-type logical operators, as shown for codes with $L = 15, 17, 19$ in Figure 13. For $L = 15$, the Z-type logical forms a square wave that traverses the lattice. Considering a 3×3 unit cell in the bulk of the lattice, we see that the logical operator has support on one third of the qubits. There is an equivalent Z-type logical operator obtained by a 90° rotation, related by a Z-type stabilizer. There are no other Z-type stabilizers or logical operators for this code (the dimension of $\ker(\mathbf{H})$ is 2). Both the $L = 17$ and $L = 19$ codes are tiled by a 3×3 unit cell with the same structure, but with different boundaries for the two codes. Within a 3×3 unit cell of these two codes, the Z-type logical operator has nontrivial support on 5/9 qubits. For both $L = 17$ and $L = 19$ there is only one Z-type logical operator (which is symmetric under 90° rotations) and no Z-type stabilizers. For all three of these codes, the structure of the Z-type logical operator and the code itself are periodic, both horizontally and vertically, with a period of 6. Therefore, adding 6 columns (or rows) to the lattice leaves the structure of the Z-type logical operator unchanged. As a result, the Z-type logical operators for $L = 15$, $L = 17$ and $L = 19$ generalize for all $L > 6$. This structure and periodicity of the Z-type logical operators explains the data in Figure 12, and the convergence of d_Z/n to 5/9 and 1/3. Therefore, at infinite bias, deforming the boundaries degrades performance relative to the XY surface code (reducing d_Z from n to $5n/9$ or $n/3$),

however we expect performance to improve substantially for noise with (even very large) finite bias.

As discussed in Section III D, we expect the logical failure rate associated with the fragile spatial boundary errors of the XY surface code to decay as $O(p^{\sqrt{n}/2+O(1)}/\sqrt{\eta}$ far below threshold. By deforming the boundaries, we expect string-like errors along the spatial boundaries to occur with probability $O(p^{\sqrt{n}/2}\eta^{-\sqrt{n}/4})$. On the other hand, by deforming the boundaries we now have pure Z-type errors occurring with probability $O(p^{cn/2})$, where here $cn = d_Z \geq n/3$ is the weight of the Z-type logical operator; see Figure 12. This $O(p^{cn/2})$ scaling is worse than the $O(p^{n/2})$ scaling of Z-type logical failures in the XY surface code (with undeformed boundaries), however for the pure Z-type errors to dominate we would require extremely high biases for any reasonable choice of p and n. Note that this analysis has only considered a few specific failure mechanisms, and a more detailed analysis of other failure mechanisms (as well as a consideration of entropic contributions to the error rate and circuit-level simulations) will be crucial to better understand and quantify the potential improvement.

VI. DISCUSSION

In this work, we have investigated the performance of the XY surface code for fault-tolerant quantum computation in the presence of biased noise. Although the XY surface code has a Z distance of n, we have identified string-like failure mechanisms, which we refer to as fragile boundary errors, that can occur at temporal boundaries (during logical state preparation and measurement), or at spatial boundaries at finite bias. These fragile boundary errors consist of $O(\sqrt{n})$ Z errors and $O(1)$ X or Y errors, and will likely dominate over errors due to the weight n Z-type logical in most realistic settings. We introduced a decoder, called belief-matching, which we show has good performance for handling biased circuit-level noise, and used it to benchmark the performance of the XY surface code compared to the CSS surface code, for which the lattice dimensions can be tailored to the bias.

We expect our belief-matching decoder will find other uses. Previous work developing decoders that handle correlations between matching graph edges in the surface code have mostly assumed perfect syndrome measurements or a phenomenological error model [11, 36–41]. However, the correlated MWPM decoder of Ref. [34] also handles errors in the syndrome extraction circuit, and has been shown to offer improved performance relative to an uncorrelated MWPM decoder [34, 35]. Rather than using BP, the correlated MWPM decoder runs MWPM twice, using hand-crafted rules to update edge-weights between the first and second use of MWPM [34]. Since the edge weight update rules of Ref. [34] consider each pair of correlated edges in isolation, unlike BP which uses the entire circuit-level Tanner graph, we might expect belief-matching to have superior decoding performance. Although belief-matching has good performance, we have shown that it does not match the performance of the computationally expensive maximum-likelihood decoder. Therefore, future work could consider alternatives to belief-matching, with the hope of finding an efficient decoder with improved decoding performance that might lead to a more favourable qubit overhead for the XY surface code below threshold.

A natural extension of belief-matching would be to use a weighted Union-Find decoder [16, 42] instead of a minimum-weight perfect matching decoder. Making this direct substitution would lead to a decoder with almost-linear runtime with the potential to maintain improved performance relative to the more computationally intensive uncorrelated minimum-weight perfect matching decoder. Furthermore, since both belief propagation and Union-Find are comparatively simple algorithms, our proposed modification may be amenable to efficient implementations in hardware [43].

There are other proposals for handling biased noise which we have not considered in this work. The XZZX surface code is a promising candidate, which has been shown to achieve very high thresholds in the presence of biased noise [8, 9]. However, optimizing the dimensions of the XZZX surface code requires using an unrotated geometry, which requires $2\times$ more qubits than the CSS surface code to achieve the same distance. Another option is to use the subsystem surface code (SSC) [44] with schedule-induced gauge-fixing, which has also been shown to have high thresholds for biased circuit-level noise [10]. The SSC uses $1.75\times$ more qubits than the CSS surface code to achieve the same distance (assuming one ancilla per gauge operator), but may be easier to build owing to its weight-three checks, which should reduce crosstalk. Importantly, unlike the XY surface code, the aspect ratios of the CSS, XZZX and subsystem surface codes can all be optimized in the presence of bias, which we have shown is highly desirable for reducing qubit overheads. While the XZZX and subsystem surface codes both offer improved performance compared to the CSS surface code near threshold for biased circuit-level noise [9, 10], a more detailed analysis will be required to assess whether this also translates into a reduced qubit overhead for a noise regime of practical interest below threshold. The CSS, XY and XZZX surface codes all fall within the broader family Clifford-deformed surface codes [45], which provide even more flexibility for tailoring the surface code to the noise bias, and further work is required to investigate these codes in a fault-tolerant setting. Finally, for architectures with improved qubit connectivity, it is possible that bias-tailored quantum LDPC codes will offer a further reduction in qubit overhead [25].

VII. ACKNOWLEDGEMENTS

ETC made his technical contributions while at Amazon Web Services. OH would like to thank Christopher Chamberland and Nikolas Breuckmann for insightful discussions about biased-noise circuit-level simulations and belief propagation, respectively. We thank Ben Brown for comments on an earlier draft.

[1] P. Aliferis and J. Preskill, Fault-tolerant quantum computation against biased noise, Physical Review A **78**, 052331 (2008).
[2] 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).
[3] J. Guillaud and M. Mirrahimi, Repetition cat qubits for fault-tolerant quantum computation, Physical Review X **9**, 041053 (2019).
[4] C. Chamberland, K. Noh, P. Arrangoiz-Arriola, E. T. Campbell, C. T. Hann, J. Iverson, H. Putterman, T. C. Bohdanowicz, S. T. Flammia, A. Keller, *et al.*, Building a fault-tolerant quantum computer using concatenated cat codes, arXiv preprint arXiv:2012.04108 (2020).
[5] D. K. Tuckett, S. D. Bartlett, and S. T. Flammia, Ultrahigh error threshold for surface codes with biased noise, Physical review letters **120**, 050505 (2018).
[6] D. K. Tuckett, A. S. Darmawan, C. T. Chubb, S. Bravyi, S. D. Bartlett, and S. T. Flammia, Tailoring surface codes for highly biased noise, Physical Review X **9**, 041031 (2019).
[7] D. K. Tuckett, S. D. Bartlett, S. T. Flammia, and B. J. Brown, Fault-tolerant thresholds for the surface code in excess of 5% under biased noise, Physical review letters **124**, 130501 (2020).
[8] J. P. Bonilla-Ataides, D. K. Tuckett, S. D. Bartlett, S. T. Flammia, and B. J. Brown, The xzzx surface code, arXiv e-prints , arXiv (2020).
[9] A. S. Darmawan, B. J. Brown, A. L. Grimsmo, D. K. Tuckett, and S. Puri, Practical quantum error correction with the xzzx code and kerr-cat qubits, arXiv preprint arXiv:2104.09539 (2021).
[10] O. Higgott and N. P. Breuckmann, Subsystem codes with high thresholds by gauge fixing and reduced qubit overhead, Physical Review X **11**, 031039 (2021).
[11] B. Criger and I. Ashraf, Multi-path summation for decoding 2d topological codes, Quantum **2**, 102 (2018).
[12] C. Horsman, A. G. Fowler, S. Devitt, and R. Van Meter, Surface code quantum computing by lattice surgery, New Journal of Physics **14**, 123011 (2012).
[13] D. Litinski and F. von Oppen, Lattice surgery with a twist: Simplifying clifford gates of surface codes, Quantum **2**, 62 (2018).
[14] D. Litinski, A game of surface codes: Large-scale quantum computing with lattice surgery, Quantum **3**, 128 (2019).
[15] C. Chamberland and E. T. Campbell, Universal quantum computing with twist-free and temporally encoded lattice surgery, arXiv preprint arXiv:2109.02746 (2021).
[16] N. Delfosse and N. H. Nickerson, Almost-linear time decoding algorithm for topological codes, arXiv preprint arXiv:1709.06218 (2017).
[17] S. Bravyi, M. Suchara, and A. Vargo, Efficient algorithms for maximum likelihood decoding in the surface code, Physical Review A **90**, 032326 (2014).
[18] D. J. MacKay and R. M. Neal, Near shannon limit performance of low density parity check codes, Electronics letters **32**, 1645 (1996).
[19] D. J. MacKay and D. J. Mac Kay, *Information theory, inference and learning algorithms* (Cambridge university press, 2003).
[20] J. Chen, A. Dholakia, E. Eleftheriou, M. P. Fossorier, and X.-Y. Hu, Reduced-complexity decoding of ldpc codes, IEEE transactions on communications **53**, 1288 (2005).
[21] D. Poulin and Y. Chung, On the iterative decoding of sparse quantum codes, arXiv preprint arXiv:0801.1241 (2008).
[22] M. P. Fossorier, Iterative reliability-based decoding of low-density parity check codes, IEEE Journal on selected Areas in Communications **19**, 908 (2001).
[23] P. Panteleev and G. Kalachev, Degenerate quantum ldpc codes with good finite length performance, arXiv preprint arXiv:1904.02703 (2019).
[24] J. Roffe, D. R. White, S. Burton, and E. Campbell, Decoding across the quantum low-density parity-check code landscape, Physical Review Research **2**, 043423 (2020).
[25] J. Roffe, L. Z. Cohen, A. O. Quintivalle, D. Chandra, and E. T. Campbell, Bias-tailored quantum ldpc codes, arXiv preprint arXiv:2202.01702 (2022).
[26] C. Gidney, Stim: a fast stabilizer circuit simulator, arXiv preprint arXiv:2103.02202 (2021).
[27] E. H. Chen, T. J. Yoder, Y. Kim, N. Sundaresan, S. Srinivasan, M. Li, A. D. Córcoles, A. W. Cross, and M. Takita, Calibrated decoders for experimental quantum error correction, arXiv preprint arXiv:2110.04285 (2021).
[28] R. Chao, M. E. Beverland, N. Delfosse, and J. Haah, Optimization of the surface code design for majorana-based qubits, Quantum **4**, 352 (2020).
[29] T. C. Bohdanowicz, Chapter 3: Tensor networks for exact and approximate maximum likelihood decoding, Thesis, California Institute of Technology (2021).
[30] D. Bacon, S. T. Flammia, A. W. Harrow, and J. Shi, Sparse quantum codes from quantum circuits, IEEE Transactions on Information Theory **63**, 2464 (2017).
[31] O. Higgott, PyMatching: A python package for decoding quantum codes with minimum-weight perfect matching, arXiv preprint arXiv:2105.13082 (2021).
[32] The parameters p and η describing the biased circuit-level noise can be alternatively defined as follows: p is the probability of any error at the given location, whereas η is the ratio of the probabilities for Z errors and any other errors. For instance, for single-qubit locations we would have $p = p_X + p_Y + p_Z$ and $\eta = p_Z/(p - p_Z)$, where p_P is the probability of a single-qubit P error.
[33] G. Torlai and M. Fishman, PastaQ: A package for simulation, tomography and analysis of quantum computers (2020).
[34] A. G. Fowler, Optimal complexity correction of correlated errors in the surface code, arXiv preprint

arXiv:1310.0863 (2013).
[35] C. Gidney, M. Newman, A. Fowler, and M. Broughton, A fault-tolerant honeycomb memory, arXiv preprint arXiv:2108.10457 (2021).
[36] G. Duclos-Cianci and D. Poulin, Fast decoders for topological quantum codes, Physical review letters **104**, 050504 (2010).
[37] G. Duclos-Cianci and D. Poulin, A renormalization group decoding algorithm for topological quantum codes, in *2010 IEEE Information Theory Workshop* (IEEE, 2010) pp. 1–5.
[38] G. Duclos-Cianci and D. Poulin, Fault-tolerant renormalization group decoder for abelian topological codes, arXiv preprint arXiv:1304.6100 (2013).
[39] J. R. Wootton and D. Loss, High threshold error correction for the surface code, Physical review letters **109**, 160503 (2012).
[40] A. Hutter, J. R. Wootton, and D. Loss, Efficient markov chain monte carlo algorithm for the surface code, Physical Review A **89**, 022326 (2014).
[41] H. Bombin, R. S. Andrist, M. Ohzeki, H. G. Katzgraber, and M. A. Martin-Delgado, Strong resilience of topological codes to depolarization, Physical Review X **2**, 021004 (2012).
[42] S. Huang, M. Newman, and K. R. Brown, Fault-tolerant weighted union-find decoding on the toric code, Physical Review A **102**, 012419 (2020).
[43] P. Das, C. A. Pattison, S. Manne, D. Carmean, K. Svore, M. Qureshi, and N. Delfosse, A scalable decoder micro-architecture for fault-tolerant quantum computing, arXiv preprint arXiv:2001.06598 (2020).
[44] S. Bravyi, G. Duclos-Cianci, D. Poulin, and M. Suchara, Subsystem surface codes with three-qubit check operators, arXiv preprint arXiv:1207.1443 (2012).
[45] A. Dua, A. Kubica, L. Jiang, S. T. Flammia, and M. J. Gullans, Clifford-deformed surface codes, arXiv preprint arXiv:2201.07802 (2022).

Appendix A: Additional numerical results

We expect that belief-matching will find useful applications beyond the XY surface code, since it provides a general method of handling hyperedges in the decoder graph for circuit-level noise, and is applicable to any code for which these hyperedges can be decomposed into edges. In Figure 14 we show the performance of belief-matching for the square, CSS surface code for circuit-level depolarising noise ($\eta = 1$), and compare its performance to that of an uncorrelated MWPM decoder. We find that belief-matching increases the threshold from 0.817(5)% to 0.940(3)%, a 1.15× improvement. It would be interesting to compare the performance of belief-matching to the *correlated* MWPM decoder of Ref. [34], which uses hand-crafted update rules between two runs of MWPM to take advantage of correlations between the matching graphs due to Y noise.

In Figure 15, we show the performance of the rectangular CSS surface code below threshold for various code distances. This is a subset of the data used to fit to the ansätze of Equation (3) and Equation (4). In Figure 16, we also show the optimal aspect ratios obtained using these fitted ansätze for a target logical failure rate of $p_{\log} = 10^{-12}$, such that the X and Z logical failure rates are equal.

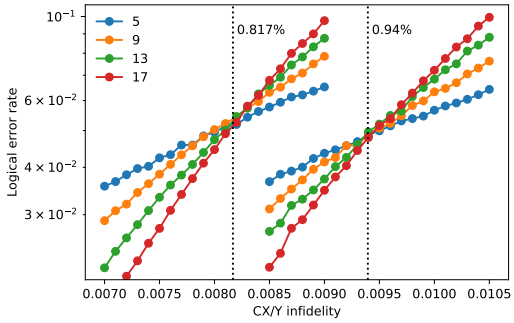


FIG. 14. Threshold of the square, CSS surface code using MWPM (left) and belief-matching (right) with $\eta = 1$ depolarising noise.

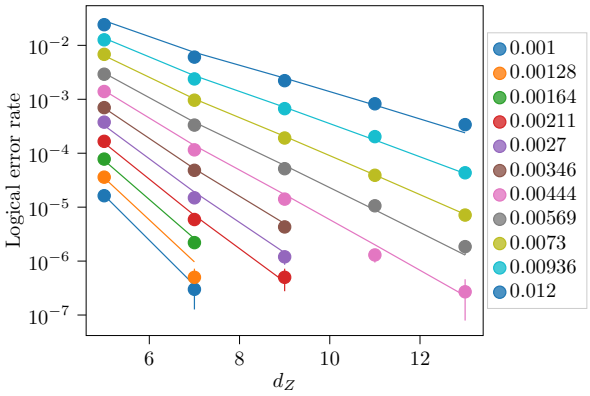


FIG. 15. Below threshold scaling of the rectangular, CSS surface code. Using an X distance of 7 (X axis specifies Z distance), and $\max(d_X, d_Z)$ rounds of stabilizer measurements.

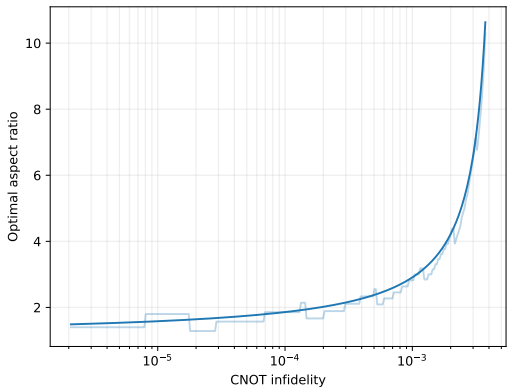


FIG. 16. Optimal aspect ratio for the rectangular, CSS surface code as a function of CNOT infidelity, for $\eta = 100$ and a target logical failure rate of $p_{\log} = 10^{-12}$. For the translucent line, code distances were restricted to odd integers. These aspect ratios were calculated by setting $p_{\log}^X = p_{\log}^Z = p_{\log}/2$ for the fitted ansätze in Equation (3) and Equation (4).