

Rare Event Simulation of Quantum Error-Correcting Circuits

Carolyn Mayer¹, Anand Ganti², Uzoma Onunkwo², Tzvetan Metodi³, Benjamin Anker^{4,5}, and Jacek Skrzyszewski⁶

¹ Quantum Computer Science 1, Sandia National Laboratories, Albuquerque, NM, USA

² Cyber Security Initiatives, Sandia National Laboratories, Albuquerque, NM, USA

³ Quantum Computer Science, Sandia National Laboratories, Albuquerque, NM, USA

⁴ Center for Quantum Information and Control, University of New Mexico, Albuquerque, NM, USA

⁵ Electrical and Computer Engineering, University of New Mexico, Albuquerque, NM, USA

⁶ Jump Trading Group, 1 London Wall, Barbican, London, UK

We describe a practical approach for accessing the logical failure rates of quantum error-correcting (QEC) circuits under low physical (component) failure rate regimes. Standard Monte Carlo is often the de facto approach for studying the failure rates of quantum circuits. However, in the study of fault-tolerant error-correcting circuits, the ability to extend this approach to low physical failure rates is limited. In particular, the use of Monte Carlo to access circuits that are relatively large or have high correcting power becomes more difficult as we lower the input failure rates of the individual components (gates) in the circuit. For these reasons, many simulations studying the circuit model go no lower than end-to-end logical failure rates in the 10^{-6} regime.

In this report, we outline an approach that borrows from earlier work by Bravyi and Vargo [6] to the more complex circuit noise model. Earlier works studied both the capacity and phenomenological noise models [3, 5, 6], but the work is insufficient for generating similar simulations in the circuit-noise model. To the best of our knowledge, our team is the first to develop a full prescription of the rare event simulation by splitting technique for the circuit-based noise model. We have also generated promising results that are confirmed by standard Monte Carlo simulation under an accessible regime. This work shows that we can access noise in the circuit-model prescription of quantum error-correcting code to failure rates below 10^{-20} regime.

Carolyn Mayer: cdmayer@sandia.gov

Anand Ganti: aganti@sandia.gov

Uzoma Onunkwo: uonunkw@sandia.gov

Tzvetan Metodi: tmetod@sandia.gov

Benjamin Anker: banker@unm.edu

1 Introduction

In recent years, strong interest has grown in the area of quantum computing. This has been made apparent by the amount of research funding and commercial sectors focusing on advancing the science for realizing quantum computing systems. These interests are motivated by opportunities to better arrive at solutions to challenging technological problems in the areas of pharmaceutical research, cybersecurity, and quantum chemistry to name a few.

However, the advancement towards quantum computing is deemed by most to be unattainable without quantum error correction and fault-tolerant systems. Many quantum computing paradigms rely on qubits, the basis of quantum information, which are quite unreliable. Current state-of-the-art systems are demonstrating systems with hundreds to thousands of qubits, capable of performing thousands of primitive quantum gate operations. [1, 14, 22, 26]. However, the application of quantum gates introduces sources of decoherence for these qubits, which must be protected. Quantum error correction (QEC) provides an avenue to protect qubits against environmental noise. With many of the advances over the last decade in hardware designs for quantum systems, today's systems are still only pushing towards 10^{-3} infidelity for quantum operations. However, many experts believe that infidelity rates in the 10^{-12} , so-called *teraquop*, range are needed for quantum systems to show real advantage over today's dominant classical computers. QEC, along with fault-tolerant designs, could bring such hardware designs to enable quantum computing in the *teraquop* range. Popular QEC codes, such as surface codes [9, 13, 16, 20, 21, 24, 29] and quantum LDPC codes [7, 12, 15, 27], show promise.

Simulation software has become imperative for assessing the performance of QEC designs on different codes and their associated decoders. These simulations en-

able researchers to understand the logical failure rates under different noise characteristics. Many works show simulation results under varying noise constraints that range from the code capacity noise model to the phenomenological noise model and the more realistic circuit noise model. The assessment of QEC designs usually leverages Monte Carlo style of simulations, but naïve approaches can limit the ability to assess codes in the teraquop regime. In this work, we illustrate how we can assess codes using a method pioneered by Bravyi and Vargo ([6]) on the phenomenological noise model for the punctured surface code. We improve upon their work by showing how to adapt simulations for the circuit-noise model and demonstrate a simulation feature that enables simulations to run orders of magnitude faster with a caching scheme that is naturally amenable to rare event simulations.

For the rest of this paper, we briefly introduce rare event simulation using the splitting method and describe the common noise models for assessing QEC and fault-tolerance in Section 2. In Section 3, we work out the extension of the Markov Chain Monte Carlo (MCMC) approach to the circuit noise model. The circuit noise model captures many of the effects that real quantum systems have at a level that is more representative of hardware effects than the code capacity and phenomenological noise models. In Section 4, we describe the parameters for our simulation study, which includes the case study for the popular rotated surface codes. We extensively show results from our simulations across a range of code distances and provide our insights on the agreement between the Monte Carlo and rare event simulation in the accessible regime. Finally, we conclude this work in Section 6 and state the future directions for this work in studies that have qubit leakage effects and circuits with dynamic size imposed by post-selections.

2 Splitting Method

The splitting method is a Metropolis-Hastings algorithm [8] that can much more efficiently explore a parameter space, which we adapt for the estimation of logical performance of QEC circuits under circuit-noise models. Traditionally, the assessment of logical performance of QEC circuits relies on Monte Carlo computer simulation approaches due to the many different gates involved in a QEC circuit. However, as the QEC circuit size increases or the physical error rates get lower, assessments using Monte Carlo simulations start becoming too time consuming. In fact, for low physical error rate (p) regimes (much below the pseudo-threshold of QEC circuits), standard Monte Carlo sampling becomes infeasible, as the number of samples needed for

an estimate becomes exorbitant. For example, trying to estimate logical failure rates ($\bar{p}$) in the teraquop regime would rely on order of a hundred trillion independent runs. For a strictly fault-tolerant encoded circuit, $\bar{p} = \Omega(p^{\lceil d/2 \rceil})$ as p approaches 0, where d is the distance of the $[[n, k, d]]$ code. The $[[n, k, d]]$ convention means a stabilizer code with n physical qubits, k logical qubits and minimum distance d .

To reduce the number of Monte Carlo samples needed to get an estimate, one can limit the size of gate failure sets sampled, noting that for large enough size m , the logical failure rate as a function of the physical error rate is:

$$\bar{p}(p) \approx \sum_{k=0}^m \binom{G}{k} p^k (1-p)^{G-k} \cdot \Pr(\text{logical fail} | \text{no. failing gates} = k),$$

where G is the number of quantum gates in the encoded circuit. This technique is sometimes called subset sampling [23].

For a strictly fault-tolerant circuit, the terms with $k < \lceil \frac{d}{2} \rceil$ do not contribute to the sum as they should not lead to any logical failures. For the cases with large k , i.e., $k > m$, the p^k terms become negligible and do not contribute significantly to the sum. However, as the code distance increases, many samples are required to estimate the $\Pr(\text{fail} | \text{no. failing gates} = k)$ term. At $k = \lceil d/2 \rceil$, many of the $\binom{G}{k}$ size- k sets may not lead to failure. See Figure 1 for estimates from simulation.

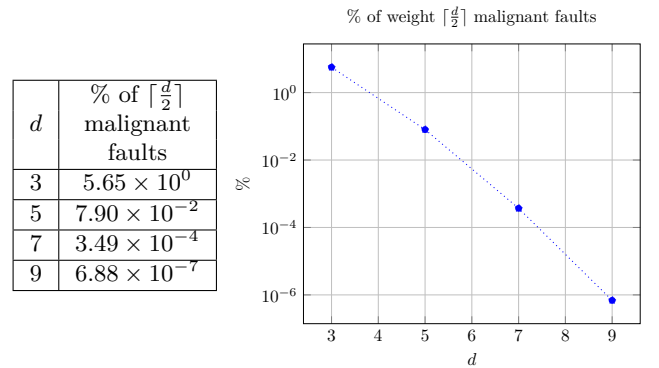


Figure 1: Estimated percentage of weight $\lceil \frac{d}{2} \rceil$ faults leading to logical failure for a distance- d rotated surface code.

Thus, estimating the probability of logical failure to sufficient accuracy given that $k = \lceil d/2 \rceil$ quantum gates failed becomes computationally challenging. Our experience is that for large code distance, and without any guarantees of strict fault tolerance, there is a tendency for the estimator to fail on this step because it assumes that $\Pr(\text{fail} | \text{no. failing gates} = k) = 0$, so as not to be

in an infinite loop cycle estimating what it assumes is zero. For instance, considering a case of 10,000 faulty quantum gates and a code distance of $d = 13$, there are potentially about 2×10^{24} samples to search for failing sets of gates of size $= \lceil 13/2 \rceil = 7$, which may number in the thousands only. That is, it would take roughly 2×10^{20} samples to get a single failing event on average, which is generally not attainable in today's classical computing systems.

The splitting method of [6] offers an alternative, although it was limited to the *code capacity* and the *phenomenological* error models (see Figure 2 for descriptions). We summarize that approach below before extending to the more complicated *circuit* model in Section 3.

Noise Models

Code capacity: Errors can occur on data qubits. Syndrome extraction is treated as an error-free process.

Phenomenological: Errors can occur on data qubits. Syndrome extraction is treated as a black box process that allows errors on the measurements.

Circuit: Syndrome extraction is modeled via a quantum circuit with one and two qubit gates. Every gate is modeled as an ideal gate composed with a Pauli noise channel. Gates fail independently, although this can be relaxed by generalizing the sampling routine. Errors can be propagated between data and ancilla qubits during syndrome extraction, e.g., through couplings with CNOT, CPHASE, or ISWAP gates.

Figure 2: Descriptions of three common error models.

Consider a monotonic sequence of parameterized physical failure rates, $p \in \{p_i\}_{i=1}^t$ and denote the corresponding output failure rates by $\{\bar{p}_i\}_{i=1}^t$. If the sequence of physical failure rates is decreasing and $\bar{p}_i$ is small, it may be easier to estimate $\bar{p}_1$ and the sequence of values $\left\{ \frac{\bar{p}_{i+1}}{\bar{p}_i} \right\}_{i=1}^{t-1}$ than it is to directly estimate the target output logical failure rate $\bar{p}_t$. This initial output failure rate, $\bar{p}_1$, can be estimated from a Monte Carlo simulation at the parameterized input rate of p_1 chosen to be relatively large so that Monte Carlo can estimate in a timely manner. In general, this splitting approach can be extended to also simulate cases where the target failure rate, $\bar{p}_t$, we wish to estimate is larger than the initial output logical rate, $\bar{p}_1$. However, the latter is of-

ten not the scenario most researchers and engineers are interested in; they often desire estimates that are closer to the teraquop regime and thus the target logical rate is much smaller than the initial failure rate estimates that Monte Carlo can achieve.

For a surface code that relies on a decoding graph, the authors prescribed a way to choose the sequence interval points for the rare event simulation using the heuristic

$$p_{i+1} = p_i \cdot 2^{\pm 1/\sqrt{w_i}}, \quad (1)$$

where $w_i = \max(d/2, p_i \cdot |\{\text{edges in decoding graph}\}|)$. The $\pm$ sign in the exponent of Equation 1 defines the direction the sequence will go as a function of where the target failure rate is with respect to the initial output failure rate. For most users, the intent would be to use rare event simulation to estimate the lower target logical failure rate, so the sequence becomes: $p_{i+1} = p_i \cdot 2^{-1/\sqrt{w_i}}$. The intuition behind choosing w_i is to start around the expected number of gates that will cause logical failures. Thus, for any quantum circuit encoded in an error-correcting code (not just the surface codes), we can adapt the splitting method so that $w_i = \max(d/2, p_i \cdot G)$, where G is the number of faulty gates in the circuit as defined before. We illustrate in the Appendix that the heuristic for picking the sequence $\left\{ \frac{p_{i+1}}{p_i} \right\}_{i=1}^{t-1}$ is really a suggestion for getting roughly the same ratios between sequence intervals, but other convenient choices would still lead to the same estimate of the target logical failure rate desired.

2.1 Notation

To describe the splitting method and prescribe the algorithm, we introduce a few notations which closely follow [6]. Let:

- Ω : sample space of all possible *events*, i.e., the powerset of all possible (gate, fault) tuples in the QEC-encoded circuit;
- $\mathcal{F}$: set of all failing (malignant) events, which are all the elements of Ω that individually lead to a logical failure in the QEC-encoded circuit;
- $\pi_i(E)$: probability of event $E \in \Omega$ at step i in the sequence of simulated p_i values;
- $\pi_{i|\mathcal{F}}$ is the conditional probability distribution over failing events defined as:

$$\pi_{i|\mathcal{F}}(E) := \begin{cases} \frac{\pi_i(E)}{\pi_i(\mathcal{F})} & \text{if } E \in \mathcal{F} \\ 0 & \text{if } E \notin \mathcal{F} \end{cases}$$

- $\mathbb{E}_{i|\mathcal{F}}(\cdot)$: expectation with respect to p_i conditioned on the failure set, $\mathcal{F}$.

2.2 Ratio Estimation

For some function $g : \mathbb{R}^+ \rightarrow \mathbb{R}^+$ such that $g(x) = x^{-1}g(x^{-1})$ and any constant $C > 0$,

$$\frac{\bar{p}_{i+1}}{\bar{p}_i} = \frac{\pi_{i+1}(\mathcal{F})}{\pi_i(\mathcal{F})} = C \frac{\mathbb{E}_{i|\mathcal{F}} \left[g \left(C \frac{\pi_i}{\pi_{i+1}} \right) \right]}{\mathbb{E}_{i+1|\mathcal{F}} \left[g \left(C^{-1} \frac{\pi_{i+1}}{\pi_i} \right) \right]}. \quad (2)$$

For a fixed number of samples, N , the function g minimizing the statistical error is $g(x) = \frac{1}{1+x}$ [2]. The expectations on the numerator and denominator will be approximated with empirical averages:

$$\begin{aligned} \mathbb{E}_{i|\mathcal{F}} \left[g \left(C \frac{\pi_i}{\pi_{i+1}} \right) \right] &\approx \mathbb{E}_{i|\mathcal{F}}^{\text{est}} \left[g \left(C \frac{\pi_i}{\pi_{i+1}} \right) \right] \\ &:= \frac{1}{N} \sum_{j=1}^N g \left(C \frac{\pi_i(E_j)}{\pi_{i+1}(E_j)} \right), \end{aligned} \quad (3)$$

where the $E_1, \dots, E_N \in \mathcal{F}$ are drawn with the probability distribution $\pi_{i|\mathcal{F}}$, found using a Metropolis routine to generate a reversible ergodic Markov chain (see 2.3).

Using the empirical estimates of the expectations, our goal is to find C such that

$$\mathbb{E}_{i|\mathcal{F}}^{\text{est}} \left[g \left(C \frac{\pi_i}{\pi_{i+1}} \right) \right] = \mathbb{E}_{i+1|\mathcal{F}}^{\text{est}} \left[g \left(C^{-1} \frac{\pi_{i+1}}{\pi_i} \right) \right]. \quad (4)$$

The C value that results in the above equality gives an estimate of the desired ratio $\frac{\bar{p}_{i+1}}{\bar{p}_i}$. This is apparent from Equation 2 because the ratio of expectations becomes unity and one is left with $\frac{\bar{p}_{i+1}}{\bar{p}_i}$ on the left-hand side and C on the right-hand side.

2.3 Reversible Markov Chain

The goal of the Metropolis routine is to produce for each i , an aperiodic and irreducible, hence ergodic, Markov Chain (RIMC) whose stationary distribution is $\pi_{i|\mathcal{F}}$. This enables us to estimate the empirical averages on the right-hand side of Equation 3. The Metropolis routine initializes a failing set $E \in \mathcal{F}$ and uses Bennett's acceptance method [2] to produce a RIMC as follows. At each iteration:

1. Select a (gate, fault) tuple e (i.e., a set of cardinality 1 in Ω) uniformly at random with respect to the unit-cardinality sets, and let E' be the symmetric difference of E, e i.e., $E \oplus e$. For the surface code and the phenomenological noise model, e can be thought of as an edge in the decoder graph (defined in Section 3); this is not the case for circuit noise models, as will be seen in Section 3.

2. Compute $q = \min \left\{ 1, \frac{\Pr(E')}{\Pr(E)} \right\}$ and perform a Bernoulli trial with success probability q . This is the acceptance step.
3. If the Bernoulli trial resulted in success and $E' \in \mathcal{F}$, output E' . Otherwise, output E .

The routine results in a Markov Chain that satisfies the detailed balance equation:

$$\frac{\Pr(E')}{\Pr(E)} = \frac{\Pr(E'|E)}{\Pr(E|E')} \quad (5)$$

and the forward chain, $E \rightarrow E'$, has the same probability as with the backward chain, $E' \rightarrow E$. Furthermore the chain is irreducible since one can start with a failing set E and arrive at another E' using the elements of $E \oplus E'$. So the Markov chain is a RIMC and its probability distribution will converge to $\pi_{i|\mathcal{F}}$.

3 Extension to Circuits

When using the circuit noise model, (gate, fault) pairs can introduce errors to the data qubits and the syndrome measurements. There exists a class of graph-based decoders such as the *minimum weight perfect matching* (MWPM) and *union-find* (UF) for surface codes that can be described on a weighted graph that we call the decoder graph. The vertices of the decoder graph are the X (Z) checks and the edges represent gate-faults or errors that result in the incident checks changing values. In the phenomenological noise model, selecting errors simply corresponds to selecting edges in the decoder graph. In the circuit noise model multiple gate-faults map to the same syndrome (edges). Due to this nature of circuit noise, we modify the state space of the MC in the original protocol in [6] from sets of edges to gate-faults. When doing so, we must modify the Metropolis routine from selecting edges in the decoding graph to selecting physical (gate, fault) pairs. Our Metropolis routine selection protocol results in a reversible MC. This is because as we show below the MC transition probabilities satisfy the detailed balance equations.

Starting with a set E of (gate, fault) pairs leading to logical failure, we select a (gate, fault) pair (g, f) uniformly at random in the circuit. If g is not in any (gate, fault) pair in E regardless of the fault, then set $E' = E \cup \{(g, f)\}$. Otherwise, $(g, h) \in E$ for some fault h . In the latter case, set $E' = (E \cup \{(g, f)\}) \setminus \{(g, h)\}$. Note that if the faults f and h are the same in this latter case, then $|E'| = |E| - 1$, where the (g, f) has been dropped. The selection process ensures E' is physically viable as opposed to a symmetric difference which would allow a single gate with two different faults. For $E' \in$

$\mathcal{F}$, the acceptance probability, $\Pr(E'|E)$, is chosen as follows:

- **Case 1:** g is not in any gate fault pair in E . Set the acceptance probability to

$$\frac{\Pr(g)}{1 - \Pr(g)} \cdot \Pr_g(f),$$

where $\Pr(g)$ is the probability of gate g failing and $\Pr_g(f)$ is the probability of the fault f occurring given that g failed.

- **Case 2:** There exists a fault h such that $(g, h) \in E$.
 1. If $f = h$, set the acceptance probability to 1.
 2. If $f \neq h$, set the acceptance probability to $\Pr_g(f)$.

In each case, the detailed balance equation is satisfied:

- **Case 1:** g is not in any gate fault pair in E , $E' = E \cup \{(g, f)\}$.

$$\begin{aligned} & \frac{\Pr(E')}{\Pr(E)} \\ &= \frac{\prod_{(i,j) \in E'} (\Pr(i) \Pr_j(j)) \prod_{(i,j) \notin E'} (1 - \Pr(i))}{\prod_{(i,j) \in E} (\Pr(i) \Pr_j(j)) \prod_{(i,j) \notin E} (1 - \Pr(i))} \\ &= \frac{\Pr(g) \Pr_g(f)}{1 - \Pr(g)} \\ &= \frac{\frac{\Pr(g)}{1 - \Pr(g)} \cdot \Pr_g(f)}{1} \\ &= \frac{\Pr(E'|E)}{\Pr(E|E')}. \end{aligned}$$

- **Case 2:** There exists a fault h such that $(g, h) \in E$.
 1. If $f = h$, $E' = E \setminus \{(g, f)\}$ or equivalently $E = E' \cup \{(g, f)\}$. This case is the reciprocal of Case 1.
 2. If $f \neq h$:

$$\frac{\Pr(E')}{\Pr(E)} = \frac{\Pr_g(f)}{\Pr_g(h)} = \frac{\Pr(E'|E)}{\Pr(E|E')}.$$

Note that as stated, these acceptance probabilities apply to modifying E by a single fault to get E' .

3.1 Extension to Leakage

The approach we have for the circuit model with Pauli noise extends to leakage naturally. The only difference is that in addition to faults we include the leakage paths. Leakage paths are the set of errors induced on the qubits that interact with a leaked qubit. The

states in the Markov chain become the sets of (gate, fault, path) leading to logical failure. We select a tuple $e := (g, f, \rho)$ uniformly at random from the set of triples in the circuit. Note a path ρ could be empty if, for instance, the gate fault does not induce leakage, i.e., the fault is purely a Pauli error. We now describe the construction of the new malignant set E' from a malignant set E and setting of q , the acceptance probability, as $\min \left\{ 1, \frac{\Pr(E')}{\Pr(E)} \right\}$. If g is not in any gate in E regardless of the fault, then set $E' = E \cup e$. Otherwise, $(g, h, \nu) \in E$ for some fault h and an optional leakage path ν . In the latter case, set $E' = (E \cup \{(g, f, \rho)\}) \setminus \{(g, h, \nu)\}$. Note that if $(f, \rho) = (h, \nu)$ then $|E'| = |E| - 1$, where the (g, f, ρ) has been dropped. The selection process ensures E' is physically viable as opposed to a symmetric difference which would allow a single gate with two different faults, an unphysical occurrence.

We consider three cases:

- **Case 1:** For $E' \in \mathcal{F}$, the acceptance probability, $\Pr(E'|E)$, is chosen as follows: g is not in any gate fault pair in E . Set the acceptance probability to

$$\frac{\Pr(g)}{1 - \Pr(g)} \cdot \Pr_g(f) \Pr_f(\rho),$$

where $\Pr(g)$ is the probability of gate g failing and $\Pr_g(f) \Pr_f(\rho)$ is the probability of the fault f occurring given that g failed.

- **Case 2:** There exists a fault h such that $(g, h, \nu) \in E$.
 1. If $(f, \rho) = (h, \nu)$, set the acceptance probability to 1.
 2. If $(f, \rho) \neq (h, \nu)$, set the acceptance probability to $\Pr_g(f) \Pr_f(\nu)$.

One can follow the analysis in the non-leakage case and show that the detailed balance equation is satisfied. Note that the addition of leakage can cause a gate selection probability to significantly increase in the proposal step. However, the acceptance step ensures that the total probability is proportional to $\Pr(g)$. For example suppose g_1 has a single fault f_1 with no fault paths due to leakage and g_2 has a single fault f_2 and M paths due to leakage. The probability of adding g_1 to a malignant set E not containing g_1 is:

$$\frac{1}{N} \frac{\Pr(g_1)}{1 - \Pr(g_1)},$$

where N is the number of triples (gate, fault, path) that can occur in the circuit. On the other hand, the probability of adding g_2 to a malignant set E not containing

g_2 is:

$$\frac{1}{N} \frac{\Pr(g_2) \sum_{\rho} \Pr_{f_2}(\rho)}{1 - \Pr(g_2)}$$

which, since $\sum_{\rho} \Pr_{f_2}(\rho) = 1$, is equivalent to

$$\frac{1}{N} \frac{\Pr(g_2)}{1 - \Pr(g_2)}$$

So the ratio of selecting g_1 versus g_2 remains unperturbed due to leakage. What is different due to leakage is the increase in the state space which results in the increase in N and hence the number of trials required for selecting a specific e .

3.2 Extensions to Post-selection

So far our Metropolis routine alters the state space by a single element. The drawback with this approach is that in the context of post-selection or logical circuits with multiple qubits, it will not necessarily lead to ergodic Markov chains. Consider an example where a quantum state is prepared using post-selection a typical approach in magic state cultivation and distillation [19, 25, 28]. If we start with a malignant set E , then the gate faults of E must lead to a zero syndrome in the post-selection part of the circuit. However, altering E by one within the ancilla state preparation and checks part of the circuit will typically lead to a non-zero syndrome and will no longer be malignant. Hence our Markov chain will lead to a collection of islands (absorption states) when we allow only single gate alterations in the Metropolis routine. In the following we present a Metropolis method that allows for multi-gate alterations and results in a reversible Markov chain. We describe the process in the context of no-leakage to simplify the notation and exposition. However, this can be easily extended to the case of leakage by adding fault-paths to the gate-faults. This comes at a higher computational complexity that grows linearly with the multi-gate selection parameter k .

We assume the Metropolis routine is initialized with a malignant set of gate-faults E .

- **Proposal Step:** Our sampling set contains sets of gate-faults up to cardinality k , i.e.,

$$S := \{\{(g_1, f_1)\}, \dots, \{(g_1, f_1), \dots, (g_k, f_k)\}\}$$

with distinct gates in the member sets that is sampled uniformly. The sampled outcome, Δ , will be used to modify the current malignant set E to form E' . Let $k_1 := |\Delta - E|$ be the number of additions, $k_2 := |\Delta \cap E|$ be the number of deletions and k_3 be the number of altered gates in Δ , i.e., they exist in E and have different faults. Note that $k = k_1 + k_2 + k_3$.

- **Acceptance Step:**

Without loss of generality, assume the gate-fault additions are $(g_1, f_1) \dots (g_{k_1}, f_{k_1})$, the deletions are $(g_{k_1+1}, f_{k_1+1}) \dots (g_{k_1+k_2}, f_{k_1+k_2})$, and the alterations in Δ are $(g_{k-k_3+1}, f_{k-k_3+1}) \dots (g_k, f_k)$. We compute the Bernoulli acceptance parameter

$$\begin{aligned} p_1 &:= \frac{\Pr(g_1)}{1 - \Pr(g_1)} \Pr_{g_1}(f_1) \dots \frac{\Pr(g_{k_1})}{1 - \Pr(g_{k_1})} \Pr_{g_{k_1}}(f_{k_1}) \\ p_2 &:= 1 \\ p_3 &:= \Pr_{g_{k-k_3+1}}(f_{k-k_3+1}) \dots \Pr_{g_k}(f_k) \\ p &:= p_1 p_2 p_3 \end{aligned}$$

where $\Pr(g_i)$ is the probability of gate g_i failing and $\Pr_{g_i}(f_i)$ is the probability of the fault f_i occurring given that g_i failed.

Recall we need to show that the above Metropolis routine satisfies the Detailed Balance Equation (5). Assume E, E' and k, k_1, k_2, k_3 are as defined above. Note that the additions for the $E \rightarrow E'$ transitions become subtractions for the $E' \rightarrow E$, and vice versa. The alterations for $E \rightarrow E'$ and $E' \rightarrow E$ are the same gates, but the faults swap from f_i to h_i . Since the gate-faults are an independent process we have

$$\begin{aligned} \frac{\Pr(E')}{\Pr(E)} &= \frac{\prod_{i=1}^{k_1} \frac{\Pr(g_i)}{1 - \Pr(g_i)} \Pr_{g_i}(f_i)}{\prod_{i=k_1+1}^{k_1+k_2} \frac{\Pr(g_i)}{1 - \Pr(g_i)} \Pr_{g_i}(f_i)} \times \prod_{i=k-k_3+1}^k \frac{\Pr_{g_i}(f_i)}{\Pr_{g_i}(h_i)} \\ &= \frac{\Pr(E'|E)}{\Pr(E|E')} \end{aligned}$$

where the last equality follows from the definition of the Metropolis routine in the acceptance step.

4 Simulation Setup

Here, we describe the setup for the simulation studies used in our analysis of the rare event error model. Our simulation studies were conducted with the $[[d^2, 1, d]]$ rotated surface code. This code is widely studied [13, 16, 21, 24] for its many advantages towards a scalable quantum computer, namely locality, simplicity of the syndrome extraction, and the existence of efficient decoder families. In our studies, we use the efficient MWPM decoder but note that other efficient decoders like the *Union-Find* decoder [10, 11] can be used as well. While our studies were for the rotated surface code under the MWPM decoder, the MCMC piece of our simulation has no real dependence on decoder selection and, as such, extends to other code and decoder configurations of interest.

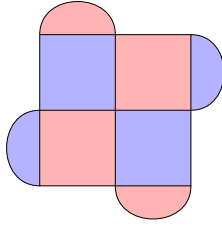


Figure 3: A distance-3 medial surface code, with X checks and Z checks. Physical data qubits are at the corners. Each color face is a check with qubits incident.

The smallest error-correcting member of the medial surface code can be described as: This code consists of X checks and Z checks as represented by the checkerboard pattern of red and blue. Figure 3 is the $[[9, 1, 3]]$ member of the family of medial surface code because it has 9 data qubits distributed at the corners of the squares and it can fix any $\frac{d-1}{2} = 1$ error. The other members of this medial surface code follow the same pattern with the grid that is required to yield a d^2 layout of data qubits, where d is the code's distance. In the circuit realization of this code, there is a need to follow a certain routine for correctness and fault tolerance of syndrome extraction. In particular, we choose a syndrome extraction routine that does not permit single faults to spread maliciously as shown in Figures 4 and 5.

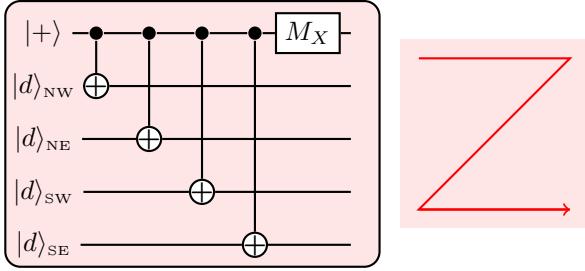


Figure 4: CNOT couplings for measuring a X -type stabilizer generator in the standard gate basis along with the direction of coupling operations on an X -type plaquette of a surface code patch. The subscript of NW, SW, NE, and SE on the data qubit represent the northwest, southwest, northeast, and southeast locations of the data with respect to the X ancilla in the grid layout.

The circuits above are repeated at each corresponding site for the X and Z syndrome bits, then repeated over $r = 2d$ rounds of syndrome extraction for fault-tolerance to the distance of the given code. There are a number of simulation software for studying the rotated surface code, including Stim [18]. In our studies, we perform simulation and evaluation of syndrome extraction in the following way:

1. For a $[[d^2, 1, d]]$ surface code, we perform noisy syn-

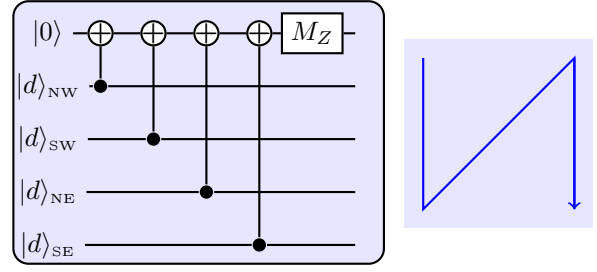


Figure 5: CNOT couplings for measuring a Z -type stabilizer generator in the standard gate basis along with the direction of coupling operations on a Z -type plaquette of a surface code patch. The subscript of NW, SW, NE, and SE on the data qubit represent the northwest, southwest, northeast, and southeast locations of the data with respect to the Z ancilla in the grid layout.

drome extractions for $2d$ rounds.

2. We perform error correction only within the first d rounds.

This approach simulates logical idle and differs from “logical T1” simulation setup of Stim. The choice of applying error correction only on the first d rounds is motivated by the need to be strictly fault-tolerant. Intuitively, having a buffer of extra d rounds enables the user to distinguish between data errors and measurement errors. The syndrome information embedded in the latter d rounds can then be used for correction after an extra d rounds of syndrome extraction.

In our circuit noise model, every quantum gate, including preparations and measurement, is assumed to have a failure rate parameterized by p , the physical gate failure rate. For our simulation results, unless noted, all gates fail with probability p . It is possible to extend to the so-called Pauli+ noise model, where gates are able to cause leakage on the input qubit(s); we do not include the Pauli+ noise model in our study.

We perform simulations in the accessible regime of standard Monte Carlo, and rare event simulations in the accessible regime and well-beyond the accessible regime of the Monte Carlo runs. The results from these scenarios enable us to assess the correctness of the rare event simulation technique in the regions of overlap, but then shows us results for the overall logical idle simulations, across all $2d$ rounds of syndrome extraction in the teraquop range.

A Monte Carlo simulation at the initial (highest) physical failure rate p_1 provides an estimate for $\bar{p}_1$ as well as a starting event for the Markov chain. We save the first malignant set (causing logical failure) during the Monte Carlo simulation as the initial event E_0 . For $i > 1$, the first event we use for p_i is the last event from the chain for p_{i-1} .

The largest runtime cost for these simulations tends to be the calls to the decoder. The Metropolis routine results in exploring the same set of malignant sets multiple times, hence caching the decoding outcome reducing repetitions of the same decoding computations, which speeds up the rare event simulation. We use a cache of recent events, containing a map from set of gate faults to the decoding outcome (i.e., failure or success). For the runs in Section 5, we use a cache size of 10M.

While a detailed study of the convergence of the Markov chains used within the rare event simulation is left as a future work, we use certain heuristics to test for convergence of the righthand side of Equation (3). In particular, for a run with N processes running their own Markov chains, we require that a process completes the following in order to provide an estimate: (1) After a user-specified minimum number J of jumps (number of times a new event is output in the Markov chain), discard the oldest event in the chain and continue for at least 10% more jumps. The rationale for discarding the older events is to account for the burn-in process. (2) Continue at least until the number of distinct faults across all the events in the chain (including discarded events) exceeds $|E_0| + \frac{d+1}{2}$.

Each process continues sampling until at least 90% of the processes ($= 0.9N$) have met both conditions, and our simulation reports the average from those completed processes. Unless otherwise specified, the results in Section 5 use $N = 20$ processes and $J = 10$ jumps.

5 Results

We present results highlighting the strengths of the rare event simulation techniques along with empirical evidence towards correctness by comparing to simple Monte Carlo simulations in accessible study regimes. We also present results to demonstrate the properties of our rare event simulation under symmetric and asymmetric circuit noise conditions, as well as under different conditions for choosing the interval points for the parameterized physical failing rates, p . The setup for the simulation results presented here follow our descriptions in Section 4. The simulations in this section were generated on an Intel Xeon server with 96 CPUs @2.7GHz and 384GB memory. The simulations were done without leakage or multi-gate fault selection.

We start by presenting the simulation results for the rotated surface code in Figure 6 under symmetric circuit noise conditions, that is, all quantum gates in the syndrome extraction circuits have the same failure probability p . Figure 6 shows the logical Z error rate (p_Z) computed using rare event simulation compared to using Monte Carlo simulations with $p \in [10^{-4}, 10^{-3}]$. These simulations are also the basis for the results

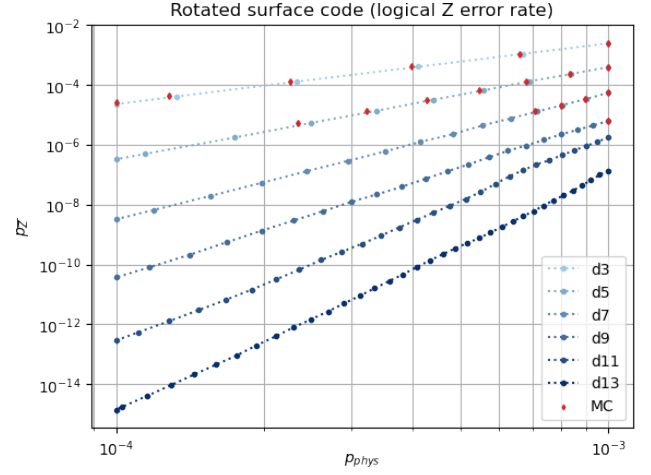


Figure 6: Logical Z error rate for a rotated surface code for $p_{phys} \in [10^{-4}, 10^{-3}]$. Estimation using Rare Event Simulation. The red diamonds show estimates using raw Monte Carlo (MC) simulation with the number of runs needed to reach at least 1,000 failures, where the estimator is the unbiased negative binomial estimator.

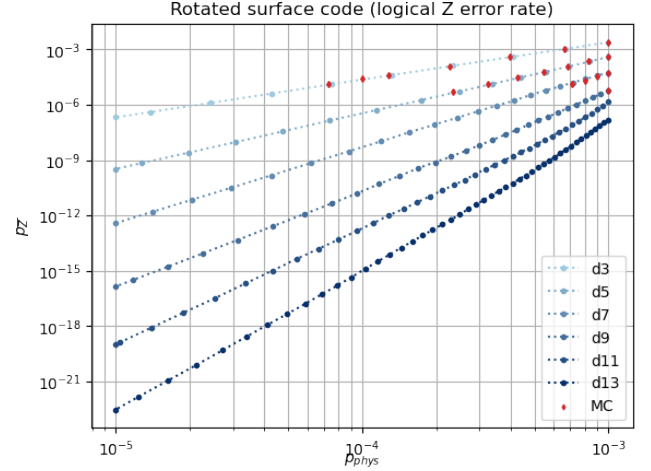


Figure 7: Logical Z error rate for a rotated surface code for $p_{phys} \in [10^{-5}, 10^{-3}]$. Estimation using rare event simulation. The red diamonds show estimates using Monte Carlo (MC) simulation with at least 1,000 failures.

shown in Figure 13 and Figure 8. The rare event simulation can still be relied on to generate valid results at lower p as shown in Figure 7. However, this would be practically impossible to generate using a standard Monte Carlo approach. For the data points that were generated for the Monte Carlo in Figures 6 and 7, the results show agreement with the rare event simulation.

While quantification of confidence for the rare event simulation estimates is left to future work, we give an empirical treatment to convergence in Figure 10 simi-

lar to a concurrent, independent work by Beverland et al [4]. Figure 10 shows the mean and standard deviation of estimates from 20 independent runs for a distance 7 rotated surface code. The logical failure estimates at each physical failure rate depend on the estimates from every previous p_i , so the estimation error can compound for lower physical error rates.

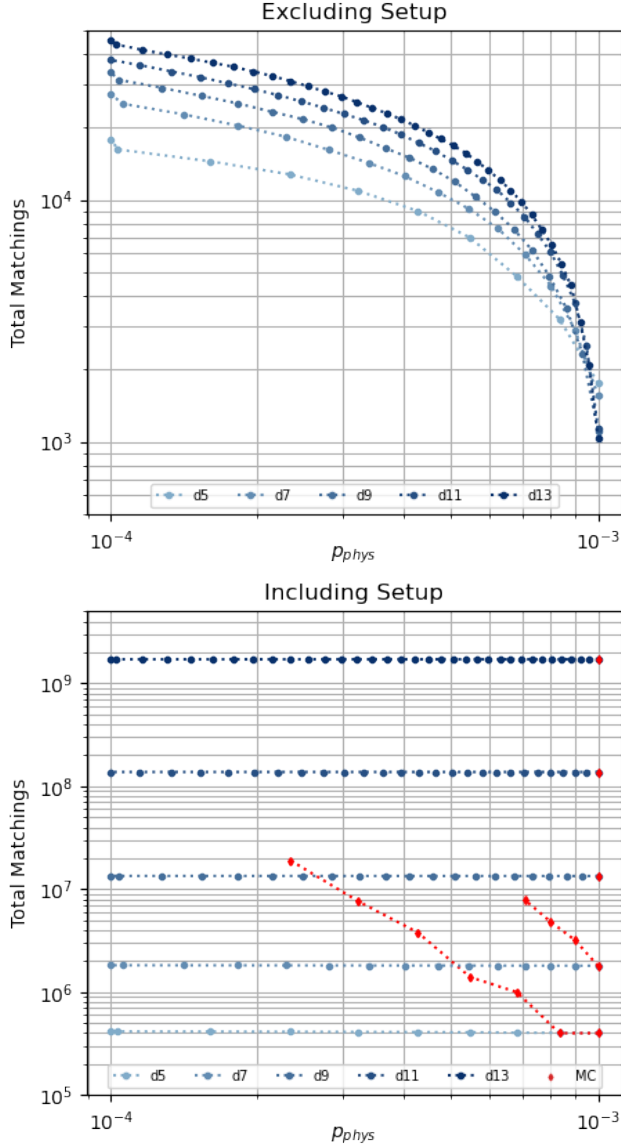


Figure 8: Total Cumulative Matchings for $p_{\overline{Z}}$ estimate (using cache). (above) Excluding the setup in which initial events and $p_{\overline{X}}$ and $p_{\overline{Z}}$ are obtained using a Monte Carlo simulation at $p_{phys} = 10^{-3}$ or (below) including the setup. The $\blacklozenge$ points show numbers of matchings to get an estimate using Monte Carlo simulation with at least 100 failures at a specified p_{phys} (not accumulated across different p_{phys}).

An interesting phenomenon with rare event simula-

tion is that as the size of the code increases, the number of p values in the heuristically-generated sequence (Equation 1) for a fixed interval of interest for p_{phys} also increases. While this potentially increases the runtime, it is generally negligible in comparison to the Monte Carlo setup time for the initial result at p_0 . Moreover, it turns out that strict adherence to the heuristically-generated sequence is not necessary. See the Appendix for further discussion.

Performing decoding, even with the MWPM decoder, is a relatively expensive step for the failure rate estimation. Compared to Monte Carlo simulations in which each sample is checked for failure, the rare event simulation relies on many fewer calls to the decoder. Figure 8 shows the cumulative number of calls to a matching (MWPM) decoder to estimate the logical failure rate for a rotated surface code at p in $[10^{-4}, 10^{-3}]$. See the Appendix for example runtimes.

The rare event simulation starts with a Monte Carlo simulation for $p = 10^{-3}$, and successive smaller physical error rates use the estimate from the previous (larger) physical error rate. We note that the frequency with which events in Ω of different cardinality are visited in the Markov chain changes with p . This is illustrated in Figure 9. At lower p , a larger portion of the events have cardinality $\lceil \frac{d}{2} \rceil$. This behavior is expected due to the fact that the expected number of failing gates goes as $G \times p$, where G is the number of quantum gates in the encoded circuit. However, when p becomes smaller, this expected number of failing gates is below the correction power of the code, $\lceil \frac{d}{2} \rceil$, so the rare event simulation steadies around events with cardinality of $\lceil \frac{d}{2} \rceil$.

For the simulations shown in the previous figures, the circuit noise was symmetrical. However, the rare event simulation easily extends to cases with asymmetrical noise, such as if qubits in certain locations of a surface code are more prone to failure than others, as might be expected in actual hardware (see, e.g., [21]). Figures 11(b) and 11(c) show logical X ($p_{\overline{X}}$) and Z error rates for a distance-11 rotated surface code in which the checks highlighted in Figure 11(a) are more likely to fail. Figure 12 shows a portion of the same plots with a comparison to estimates found using standard Monte Carlo simulations.

6 Conclusions and Future Work

6.1 Future Work

We identify several directions for future work.

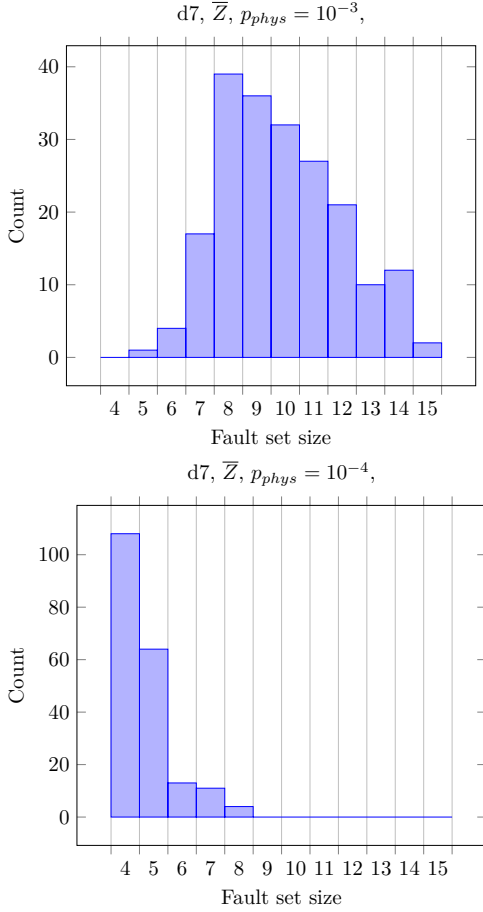


Figure 9: Count histograms of sizes of fault sets visited in estimation of $p_{\bar{Z}}$ at (top) $p_{phys} = 10^{-3}$ and (bottom) $p_{phys} = 10^{-4}$ for a distance-7 surface code. Note that at the lower p_{phys} , faults close to size $\lceil \frac{d}{2} \rceil$ account for a larger portion of events.

6.1.1 Confidence and Convergence

In the rare event simulation, the ratio C satisfying Equation 4 is estimated for each consecutive pair of physical error probabilities in the splitting sequence. Since the ratios are multiplied to estimate the logical failure rate, errors in these estimates accumulate. The matters of (1) ensuring that enough jumps have been taken for convergence of the Markov chains and (2) quantifying confidence for estimates remain open.

A diagnostic for convergence such as the Gelman-Rubin statistic [17] can provide an intuitive, computable test, but does not provide a measure of confidence while also being sensitive to iterative statistics following a long tail or with different within-chain variance.

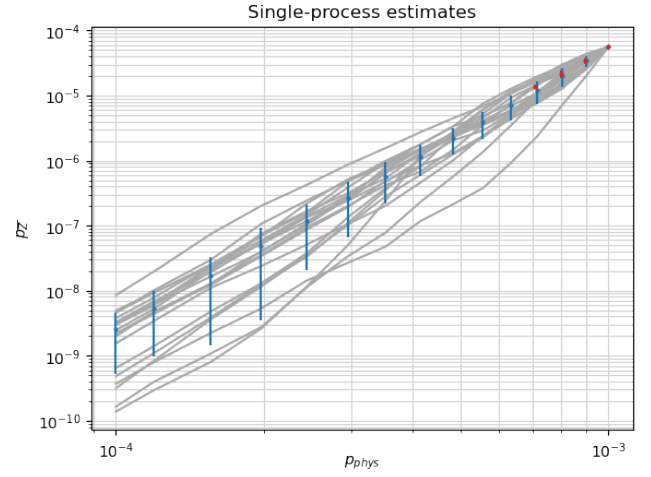


Figure 10: Logical Z error rate estimates in 20 rare event simulation runs with a single process for $d = 7$. The blue bars show the mean $\pm$ one standard deviation. The mean $\pm$ one standard deviation for 10 Monte Carlo runs with at least 100 failures each is shown in red.

6.1.2 Splitting Heuristics

To determine sequences of p values, we used the heuristic in Equation 1 chosen by [6] for its tradeoff between the number of splitting steps and statistical error. In cases with asymmetric noise, one could study a tradeoff between statistical error and the number of p values.

6.1.3 Further Simulations

In Sections 3.1 and 3.2, we described extensions to circuits with leakage and with post-selection. Implementing these extensions and providing simulated results remains as future work.

6.2 Conclusions

We have outlined an approach extending earlier work by Bravyi and Vargo [6] to the more complex circuit noise model. For our extension to the circuit model, we performed simulation for logical failure rates below 10^{-20} regime and showed agreement with Monte Carlo simulations in a more accessible ($> 10^{-6}$) regime. Our work has provided a technique for which other simulation software can be developed for studying teraquop logical failure rates deemed useful for meaningful quantum computation under realistic noise conditions.

7 Author Contributions

All authors contributed substantially to this work. No generative AI models were used directly in the genera-

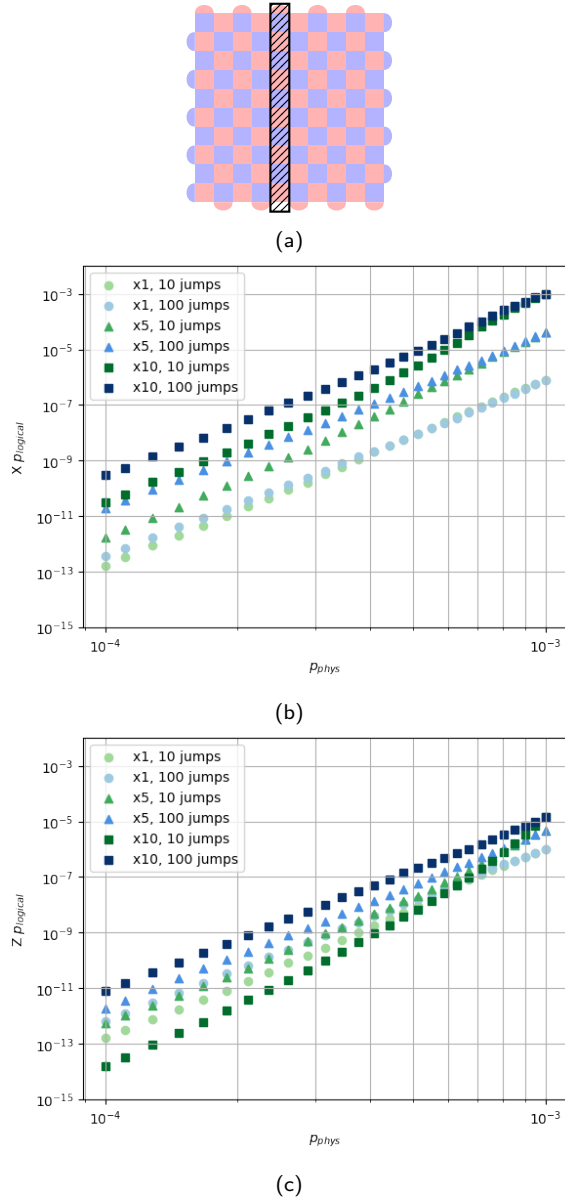


Figure 11: Logical X and logical Z error rates for a distance-11 rotated surface code with asymmetric noise using a minimum of 100 jumps (number of times a new event is output in the Markov chain) or 10 jumps in at least 18 of 20 processes before moving to the next p_{phys} . The physical error in the column marked in (a) is multiplied by the value indicated in the legend. While using 10 jumps returns results faster, doing so may not provide sufficiently accurate results.

tion, analysis, or writing of this paper.

Acknowledgments

We want to express immense gratitude to David Collins and Joseph Maurer at Sandia National Laboratories,

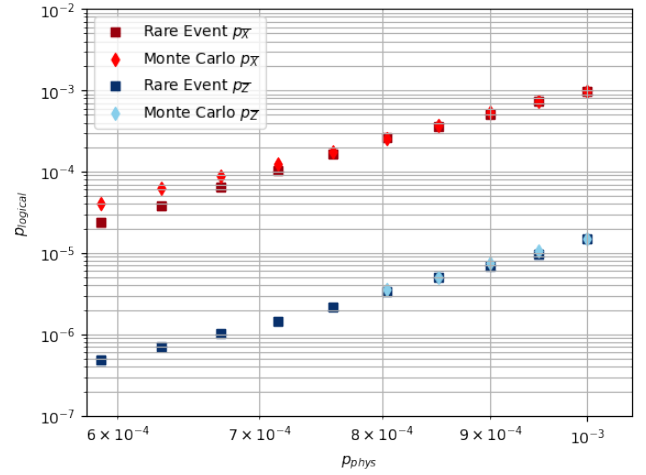


Figure 12: Part of the asymmetric noise data in Figure 11 with $10\times$ the noise in a column and ≥ 100 jumps compared to Monte Carlo estimates. The Monte Carlo processes use at least 1,000 failures to estimate $p_{\bar{x}}$ and $p_{\bar{z}}$.

who were instrumental in the base code development that was matured to enable our rare event simulations for circuit-based noise models. We also want to thank Andrew Landahl and Antonio Russo at Sandia National Laboratories for their valuable discussions and expert support of ideas as they were being developed for this work.

Sandia National Laboratories is a multi-mission laboratory managed and operated by National Technology & Engineering Solutions of Sandia, LLC (NTESS), a wholly owned subsidiary of Honeywell International Inc., for the U.S. Department of Energy's National Nuclear Security Administration (DOE/NNSA) under contract DE-NA0003525. This written work is authored by an employee of NTESS. The employee, not NTESS, owns the right, title and interest in and to the written work and is responsible for its contents. Any subjective views or opinions that might be expressed in the written work do not necessarily represent the views of the U.S. Government. The publisher acknowledges that the U.S. Government retains a non-exclusive, paid-up, irrevocable, world-wide license to publish or reproduce the published form of this written work or allow others to do so, for U.S. Government purposes. The DOE will provide public access to results of federally sponsored research in accordance with the DOE Public Access Plan.

This paper describes objective technical results and analysis. Any subjective views or opinions that might be expressed in the paper do not necessarily represent the views of the U.S. Department of Energy or the United States Government.

References

- [1] Muhammad AbuGhanem. IBM quantum computers: evolution, performance, and future directions. *The Journal of Supercomputing*, 81(5):687, 2025. DOI: <https://doi.org/10.1007/s11227-025-07047-7>.
- [2] Charles H Bennett. Efficient estimation of free energy differences from Monte Carlo data. *Journal of Computational Physics*, 22(2):245–268, 1976. ISSN 0021-9991. DOI: [https://doi.org/10.1016/0021-9991\(76\)90078-4](https://doi.org/10.1016/0021-9991(76)90078-4).
- [3] Michael E Beverland, Benjamin J Brown, Michael J Kastoryano, and Quentin Marolleau. The role of entropy in topological quantum error correction. *Journal of Statistical Mechanics: Theory and Experiment*, 2019(7):073404, jul 2019. DOI: <https://doi.org/10.1088/1742-5468/ab25de>.
- [4] Michael E Beverland, Malcolm Carroll, Andrew W Cross, and Theodore J Yoder. Fail fast: techniques to probe rare events in quantum error correction. *arXiv preprint arXiv:2511.15177*, 2025. DOI: <https://doi.org/10.48550/arXiv.2511.15177>.
- [5] J Pablo Bonilla Ataides, David K Tuckett, Stephen D Bartlett, Steven T Flammia, and Benjamin J Brown. The XZZX surface code. *Nature communications*, 12(1):2172, 2021. DOI: <https://doi.org/10.1038/s41467-021-22274-1>.
- [6] Sergey Bravyi and Alexander Vargo. Simulation of rare events in quantum error correction. *Phys. Rev. A*, 88:062308, Dec 2013. DOI: <https://doi.org/10.1103/PhysRevA.88.062308>.
- [7] Nikolas P Breuckmann and Jens Niklas Eberhardt. Quantum low-density parity-check codes. *PRX Quantum*, 2(4):040101, 2021. DOI: <https://doi.org/10.1103/PRXQuantum.2.040101>.
- [8] Siddhartha Chib and Edward Greenberg. Understanding the Metropolis-Hastings algorithm. *The American Statistician*, 49(4):327–335, 1995. DOI: <https://doi.org/10.1080/00031305.1995.10476177>.
- [9] Nicolas Delfosse. Decoding color codes by projection onto surface codes. *Physical Review A—Atomic, Molecular, and Optical Physics*, 89(1):012317, 2014. DOI: <https://doi.org/10.1103/PhysRevA.89.012317>.
- [10] Nicolas Delfosse and Matthew B. Hastings. Union-Find Decoders For Homological Product Codes. *Quantum*, 5:406, March 2021. ISSN 2521-327X. DOI: <https://doi.org/10.22331/q-2021-03-10-406>.
- [11] Nicolas Delfosse and Naomi H Nickerson. Almost-linear time decoding algorithm for topological codes. *Quantum*, 5:595, 2021. DOI: <https://doi.org/10.22331/q-2021-12-02-595>.
- [12] Nicolas Delfosse, Vivien Londe, and Michael E Beverland. Toward a union-find decoder for quantum LDPC codes. *IEEE Transactions on Information Theory*, 68(5):3187–3199, 2022. DOI: <https://doi.org/10.1109/TIT.2022.3143452>.
- [13] Eric Dennis, Alexei Kitaev, Andrew Landahl, and John Preskill. Topological quantum memory. *Journal of Mathematical Physics*, 43(9):4452–4505, 2002. DOI: <https://doi.org/10.1063/1.1499754>.
- [14] Alberto Di Meglio, Karl Jansen, Ivano Tavernelli, Constantia Alexandrou, Srinivasan Arunachalam, Christian W Bauer, Kerstin Borrás, Stefano Carrazza, Arianna Crippa, Vincent Croft, et al. Quantum computing for high-energy physics: State of the art and challenges. *PRX quantum*, 5(3):037001, 2024. DOI: <https://doi.org/10.1103/PRXQuantum.5.037001>.
- [15] Irit Dinur, Min-Hsiu Hsieh, Ting-Chun Lin, and Thomas Vidick. Good quantum LDPC codes with linear time decoders. In *Proceedings of the 55th annual ACM symposium on theory of computing*, pages 905–918, 2023. DOI: <https://doi.org/10.1145/3564246.3585101>.
- [16] Austin G. Fowler, Matteo Mariantoni, John M. Martinis, and Andrew N. Cleland. Surface codes: Towards practical large-scale quantum computation. *Phys. Rev. A*, 86:032324, Sep 2012. DOI: <https://doi.org/10.1103/PhysRevA.86.032324>.
- [17] Andrew Gelman and Donald B. Rubin. Inference from Iterative Simulation Using Multiple Sequences. *Statistical Science*, 7(4):457 – 472, 1992. DOI: <https://doi.org/10.1214/ss/1177011136>.
- [18] Craig Gidney. Stim: a fast stabilizer circuit simulator. *Quantum*, 5:497, July 2021. ISSN 2521-327X. DOI: <https://doi.org/10.22331/q-2021-07-06-497>.
- [19] Craig Gidney, Noah ShuTTY, and Cody Jones. Magic state cultivation: growing T states as cheap as cnot gates. *arXiv preprint arXiv:2409.17595*, 2024. DOI: <https://doi.org/10.48550/arXiv.2409.17595>.
- [20] Craig Gidney, Michael Newman, Peter Brooks, and Cody Jones. Yoked surface codes. *Nature Communications*, 16(1):4498, 2025. DOI: <https://doi.org/10.1038/s41467-025-59714-1>.
- [21] Google Quantum AI. Suppressing quantum errors by scaling a surface code logical qubit. *Nature*, 614(7949):676–681, 2023. DOI: <https://doi.org/10.1038/s41586-022-05434-1>.
- [22] Google Quantum AI and Collaborators. Quantum error correction below the surface code threshold. *Nature*, 638(8052):920–926, 2025. DOI: <https://doi.org/10.1038/s41586-024-08449-y>.
- [23] Sascha Heußen, Don Winter, Manuel Rispler, and Markus Müller. Dynamical subset sampling of quantum error-correcting protocols. *Physi-*

- cal Review Research*, 6(1):013177, 2024. DOI: <https://doi.org/10.1103/PhysRevResearch.6.013177>.
- [24] Dominic Horsman, Austin G Fowler, Simon Devitt, and Rodney Van Meter. Surface code quantum computing by lattice surgery. *New Journal of Physics*, 14(12):123011, 2012. DOI: <https://doi.org/10.1088/1367-2630/14/12/123011>.
 - [25] Ying Li. A magic state’s fidelity can be superior to the operations that created it. *New Journal of Physics*, 17(2):023037, 2015. DOI: <https://doi.org/10.1088/1367-2630/17/2/023037>.
 - [26] Hannah J Manetsch, Gyohei Nomura, Elie Bataille, Xudong Lv, Kon H Leung, and Manuel Endres. A tweezer array with 6,100 highly coherent atomic qubits. *Nature*, 647(8088):60–67, 2025. DOI: <https://doi.org/10.1038/s41586-025-09641-4>.
 - [27] Pavel Panteleev and Gleb Kalachev. Quantum LDPC codes with almost linear minimum distance. *IEEE Transactions on Information Theory*, 68(1):213–229, 2021. DOI: <https://doi.org/10.1109/TIT.2021.3119384>.
 - [28] Pedro Sales Rodriguez, John M Robinson, Paul Niklas Jepsen, Zhiyang He, Casey Duckering, Chen Zhao, Kai-Hsin Wu, Joseph Campo, Kevin Bagnall, Minh Kwon, et al. Experimental demonstration of logical magic state distillation. *Nature*, 645(8081):620–625, 2025. DOI: <https://doi.org/10.1038/s41586-025-09367-3>.
 - [29] Boris M Varbanov, Marc Serra-Peralta, David Byfield, and Barbara M Terhal. Neural network decoder for near-term surface-code experiments. *Physical Review Research*, 7(1):013029, 2025. DOI: <https://doi.org/10.1103/PhysRevResearch.7.013029>.

A Appendix

A breakdown of the runtimes for our implementation of the rare event simulation for odd distance surface codes is shown in Figure 13. As the distance increases, the initial setup, which includes a standard Monte Carlo simulation at the starting point where $p = 10^{-3}$ accounts for the bulk of the simulation time. Note that this starting point is

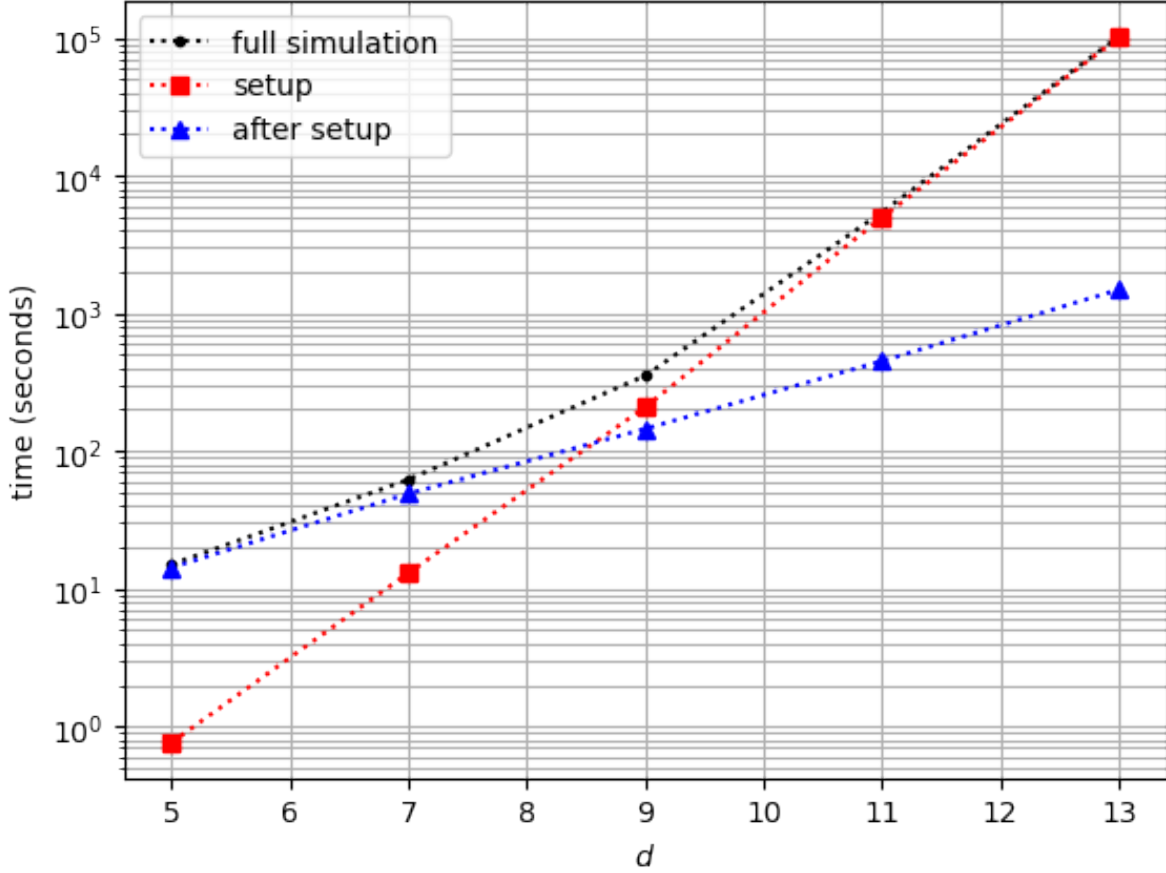


Figure 13: Cumulative time to run the simulation with $p_{phys} \in [10^{-4}, 10^{-3}]$. The setup time includes running a Monte Carlo simulation at $p_{phys} = 10^{-3}$ to generate the initial failing events and to get the initial estimates of the logical failure rates, $p_{\bar{X}}$ and $p_{\bar{Z}}$.

the easiest/fastest for the Monte Carlo simulation because fewer runs are needed to generate a high quality estimate of the initial logical failure rate for seeding the rare event simulation. For higher distance codes, starting at a larger p (chosen below the pseudo-threshold¹) may speed up the simulation. We also note that while the Monte Carlo simulation can be sped up through parallelization (e.g. using MPI or threads), our implementation of the rare event algorithm is not amenable to parallelization within a MCMC chain; however, parallelization helps with the quality of the estimator as used by convergence techniques like the Gelman-Rubin diagnostic [17].

Figure 14 shows comparisons of $p_{\bar{Z}}$ estimates using different sequences of p for the distance-5 rotated surface code, and Figure 15 shows estimates for the distance-11 rotated surface code. For distance-5, using different sequences of p yields similar estimates. For the points labeled “twice” in the plot, a point was added halfway between each point of the heuristic, Equation 1. For distance-11, there is some difference between the estimates from different splitting schemes but we suspect that this is the effect of not yet reaching the steady state distribution in the MCMC approach.

¹The pseudo-threshold for a code is the point below which the logical failure rate is below the physical gate failure rate.

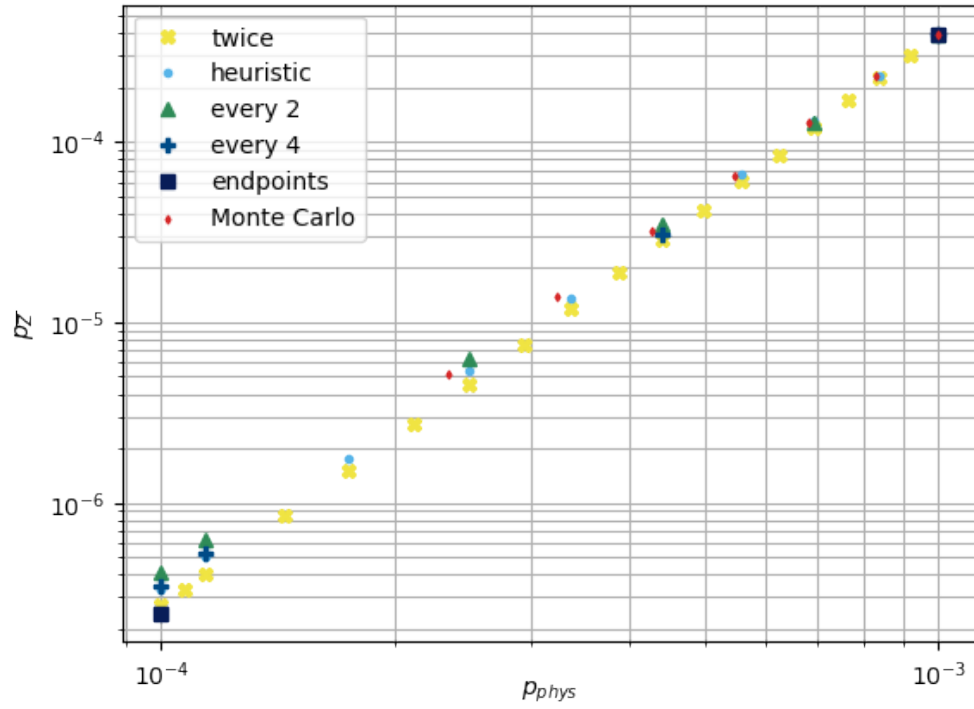


Figure 14: Logical Z error rate estimates for a distance-5 rotated surface code for $p_{phys} \in [10^{-4}, 10^{-3}]$. The legend entries refer to the points in the p_{phys} sequence relative to the splitting heuristic.

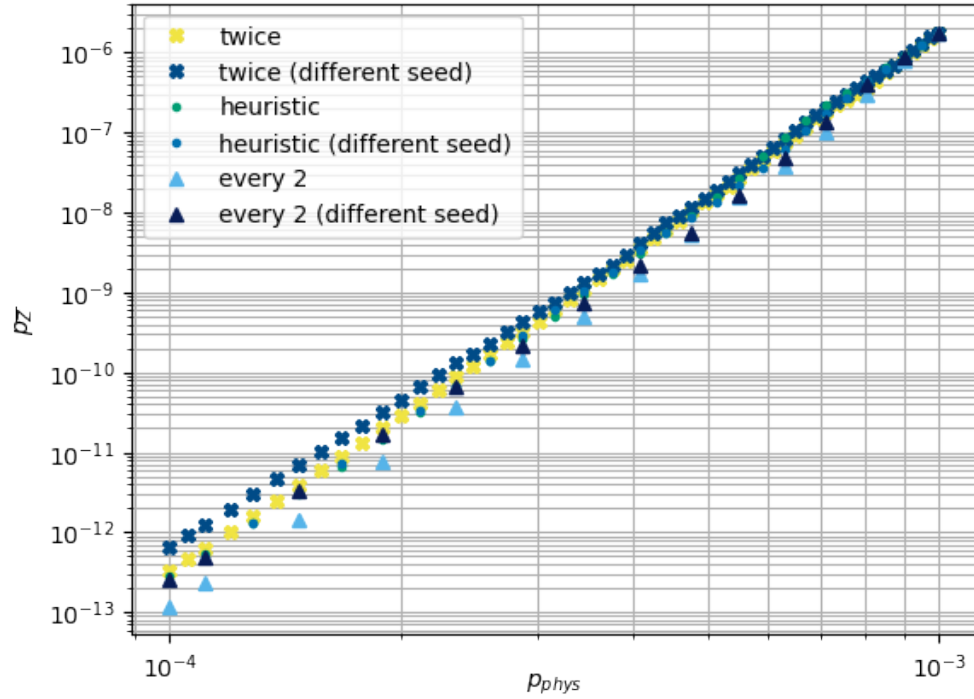


Figure 15: Logical Z error rate estimates for a distance-11 rotated surface code for $p_{phys} \in [10^{-4}, 10^{-3}]$. Estimation using rare event simulation with sequences of physical failure rates.