

Hardware-Realistic Surface-Code Simulation under Stochastic Gate Noise

Cherilyn Lee Gabriela Christen^{1*}, Nathaniel James Pacey^{1*}, Michele Grossi², Perrine Vantaloni¹, Roman Wixinger^{2†}, Paolo Da Rold³, Nicolas Macris¹, Giovanni Di Bartolomeo³, Michele Vischi³

¹*École Polytechnique Fédérale de Lausanne (EPFL), Laboratory for Statistical Mechanics of Inference in Large Systems (SMILS), Lausanne, Switzerland*

²*The European Organization for Nuclear Research (CERN), Geneva 1211, Switzerland*

³*Department of Physics, University of Trieste, Strada Costiera 11, 34151 Trieste, Italy*

Motivation

Simulating quantum circuits under realistic noise is essential for predicting how near-term hardware will behave before committing expensive device time to an experiment. This is true for noisy quantum algorithms, quantum machine-learning workflows, and quantum error-correction (QEC) protocols, whose performance depends not only on the ideal circuit, but also on the underlying physical noise process, hardware connectivity, and gate schedule [1, 2]. Simulations that neglect these effects may overestimate the useful circuit depth of an algorithm or obscure the mechanisms responsible for algorithmic failures.

In this work, we investigate how a repeated stabilizer protocol, concretely, a distance-3 rotated surface code, performs once it is exposed to hardware-calibrated stochastic gate noise, transpilation, and backend connectivity, rather than to a simplified phenomenological error model. We study this question using the stochastic noisy-gate formalism of the open-source *quantum-gates* library for Qiskit, which derives noisy gates from open-system dynamics [3]. That formalism is well suited to isolated noisy gates, but the original library was not built for dynamic, multi-cycle protocols: it lacked mid-circuit measurement, reset, intermediate state readout, and direct support for two-qubit operations on physically connected qubits that are not adjacent in statevector index order. Reaching a repeated-cycle QEC circuit therefore required extending the execution model itself. The resulting simulator is not specific to surface codes: it applies to any protocol built from repeated measurement, reset, and dynamic control flow.

This complements standard QEC simulators such as `STIM` [6], which excel at large-distance stabilizer-formalism sampling under Pauli noise but do not capture generic non-Pauli error channels or the Lindblad-equation dynamics that the noisy-gates approach targets.

Stochastic Noisy-Gate Model

The underlying noise model starts from Lindblad evolution during a driven gate, where noise during gate execution is modeled using the Lindblad formalism for Markovian open quantum systems. At the density-matrix level, the evolution is governed by

$$\frac{d\rho}{dt} = -\frac{i}{\hbar}[H, \rho] + \sum_k \left(L_k \rho L_k^\dagger - \frac{1}{2} \{L_k^\dagger L_k, \rho\} \right) \quad (1)$$

where H is the system Hamiltonian and $\{L_k\}$ are Lindblad operators describing environmental coupling.

For circuit-level simulation, this master equation is unraveled into an Itô stochastic differential equation for state evolution.

$$d|\psi_s\rangle = \left[-\frac{i}{\hbar} H_s ds + \sum_k \left(i\epsilon dW_{k,s} L_k - \frac{\epsilon^2}{2} ds L_k^\dagger L_k \right) \right] |\psi_s\rangle \quad (2)$$

Where $dW_{k,s}$ are independent Wiener increments and $\epsilon \ll 1$ sets the noise scale. These dynamics compile into an effective noisy gate [3]:

$$N_g = U_g e^\Lambda e^{\tilde{\Xi}}, \quad (3)$$

where U_g is the ideal gate, Λ is a deterministic drift term, and $\tilde{\Xi}$ the stochastic term is sampled from a covariance structure derived from the interaction-picture Lindblad operators. Ensemble averaging over trajectories recovers the Lindblad evolution, but individual trajectories need not be trace preserving. This

*Both authors contributed equally.

†Former member of CERN.

is important for repeated measurement and reset, since a mid-circuit measurement should not renormalize away norm already lost to noise.

Simulator Extensions

The key methodological step is a chunked circuit representation. Gates that can be composed without inspecting the quantum state are grouped into noisy-evolution chunks. Operations that require direct access to the state (measurement, reset, and readout) terminate the current chunk and are executed explicitly:

$$\mathcal{C} = [(\mathcal{E}_1, 0), (\mathcal{F}_1, 1), (\mathcal{E}_2, 0), (\mathcal{F}_2, 1), \dots], \quad (4)$$

where $\mathcal{E}_i$ denotes a composable noisy evolution and $\mathcal{F}_i$ denotes a state-interrupting operation. This preserves the batched-evolution structure of the library while allowing dynamic-circuit operations needed for stabilizer cycles.

Mid-circuit measurement is implemented by sampling Born probabilities from a normalized copy of the incoming state, then applying the corresponding projection while carrying forward the incoming trajectory norm. Reset is implemented as measurement followed by a conditional noisy correction. Virtual R_z gates are tracked through per-qubit phase accumulators that survive chunk boundaries, preserving Qiskit transpilation semantics [4]. Intermediate state readout is handled through labeled barriers rather than simulator-only save instructions, since barriers survive backend-aware transpilation.

We also extend connectivity handling. Some hardware backends contain physical couplings between qubits whose statevector indices are not adjacent. Rather than inserting SWAP networks solely to satisfy the simulator’s index ordering, we permute the statevector axes to bring the target qubits together, apply the two-qubit gate, and invert the permutation. On a 16-qubit circuit transpiled to *FakeBrisbane*, this permutation approach achieves comparable runtime to an index-adjacent gate, 2.69, s versus 3.12, s, while being approximately $5\times$ faster than SWAP-based routing of the same logical gate, which takes 14.49, s, as shown in Table 1. The SWAP-routed circuit requires 13 ECR gates instead of one, leading to more statevector updates and $14\times$ more tensor contractions per shot. (Table 1).

Table 1: Profiling results on a 16-qubit circuit, 100 shots on the *FakeBrisbane* backend, comparing a consecutive two-qubit gate, SWAP-routed non-consecutive gate, and permutation-based non-consecutive gate.

	Consecutive cx (4, 5)	SWAP routing cx (4, 14)	Permutation cx (4, 15)
Total time	3.12s	14.49s	2.69s
ECR gates	1	13	1
statevector calls	100	700	300
tensordot calls	1,600	22,400	1,600

Noise Scaling and Customization

For benchmarking, we introduce custom noise classes that allow both global and targeted control of backend-derived noise. A global scale factor s can rescale gate error and coherence parameters as $p \rightarrow sp$, $T_1 \rightarrow T_1/s$, $T_2 \rightarrow T_2/s$, while additional controls allow specific noise channels, qubits, or operations to be varied or disabled. This makes it possible to stress-test circuits and isolate which hardware effects most strongly affect QEC performance.

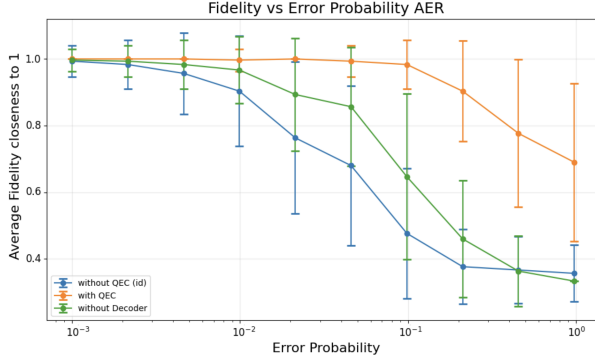
Surface-Code Benchmark

We benchmark this simulator on a rotated distance-3 surface code (17 qubits: 9 data, 8 ancilla), which stresses every mechanism above at once. Each run uses three QEC cycles and 100 shots per error probability, with fidelity evaluated on the data qubits after tracing out the ancillas.

We compare three cases: an identity-like non-QEC baseline, a surface-code circuit without decoder corrections, and a surface-code circuit corrected by a phenomenological minimum-weight matching decoder implemented with `PyMatching` [5]. We first validate the surface-code construction against Qiskit’s Aer simulator on non-transpiled circuits, using a single-qubit depolarizing channel with probability p applied

to every single-qubit gate on the data qubits only; ancilla qubit gates are left noiseless in this validation step, and p is swept over the range shown in Fig. 1(a). We then study the same logical task under backend-aware execution with the stochastic noisy-gate model, using *ScaledNoiseGates* with the same noise-scaling parameter p swept over the identical range as in panel (a), applied to the data qubits only.

(a)



(b)

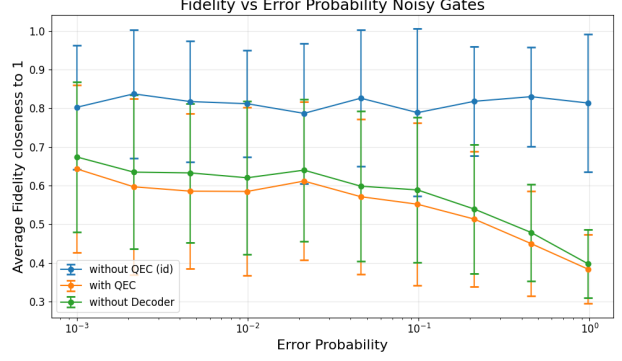


Figure 1: Distance-3 rotated surface-code benchmarks over three QEC cycles. (a) In the controlled Aer validation, decoding protects the data-qubit state below an apparent pseudo-threshold near $p \sim 10^{-2}$. (b) In the extended stochastic noisy-gates simulation, the same QEC circuit performs substantially worse than the non-QEC baseline, indicating that gate-level noise, timing, and backend execution effects can dominate the simplified decoder assumptions.

The Aer validation in Fig. 1(a) confirms that the surface-code circuit and decoder behave as expected in an idealized setting: at low error rates, the decoded circuit maintains high fidelity and outperforms both the undecoded code and the non-QEC baseline. In contrast, Fig. 1(b) shows a less favorable regime in the stochastic noisy-gates simulation. The QEC curves fall below the baseline over much of the tested range, and the decoded and undecoded cases remain close. The non-QEC baseline itself is close to flat across the swept range in panel (b).

This follows from the composition of the two circuits: the identity circuit contains only single-qubit gates, whereas the QEC circuits include repeated syndrome-extraction operations that entangle data and ancilla qubits. Although the noise scaling is applied at the level of data-qubit gates, these entangling operations allow local gate errors to propagate through the circuit and affect the multi-qubit stabilizer dynamics. Since p rescales the noise of all data-qubit gates by the same factor, this produces a much larger absolute change in fidelity for the QEC circuits than for the single-qubit-only baseline, which stays comparatively flat over the tested range.

This does not imply that the surface code itself is ineffective; rather, it shows that realistic gate-level noise, timing, transpilation, and backend-execution assumptions can invalidate a simplified phenomenological decoding model. These results motivate more hardware-aware surface-code simulation before drawing conclusions about fault-tolerant performance.

Conclusion and Future Work

In an idealized Aer setting the code shows the expected decoded advantage; under the stochastic noisy-gate model the same class of circuits can perform much worse, showing that oversimplified noise assumptions may mislead experimental design.

To make this kind of study possible, we extended the open-source *quantum-gates* library for Qiskit [3] with the execution mechanisms that repeated, dynamic-circuit protocols require: chunked execution, mid-circuit measurement and reset, transpilation-compatible intermediate readout, phase tracking, efficient handling of non-adjacent coupled qubits, and customizable noise scaling. These extensions are not specific to surface codes and can be reused directly for other repeated-measurement or dynamic-circuit protocols; the corresponding code has been contributed back to the *quantum-gates* repository.

Ongoing work focuses on reducing routing overhead, incorporating circuit-level spacetime decoding, testing larger code distances, and adding correlated two-qubit noise, then extending the same methodology to real superconducting devices through CERN QTI access and, eventually, other hardware platforms.

Code Availability

The extensions described here have been contributed to the open-source CERN-QTI *quantum-gates* library, where they are available for reuse in other repeated-measurement and dynamic-circuit protocols: github.com/CERN-QTI/quantum-gates.

References

- [1] A. G. Fowler, M. Mariantoni, J. M. Martinis, and A. N. Cleland, “Surface codes: Towards practical large-scale quantum computation,” *Physical Review A*, vol. 86, p. 032324, 2012.
- [2] Y. Tomita and K. M. Svore, “Low-distance surface codes under realistic quantum noise,” *Physical Review A*, vol. 90, p. 062320, 2014.
- [3] G. Di Bartolomeo, M. Vischi, F. Cesa, R. Wixinger, M. Grossi, S. Donadi, and A. Bassi, “Noisy gates for simulating quantum computers,” *Physical Review Research*, vol. 5, p. 043210, 2023.
- [4] A. Javadi-Abhari *et al.*, “Quantum computing with Qiskit,” *Quantum*, vol. 8, p. 1137, 2024.
- [5] O. Higgott and C. Gidney, “PyMatching: A Python package for decoding quantum error-correcting codes,” *Quantum*, vol. 5, p. 605, 2021.
- [6] C. Gidney, “Stim: a fast stabilizer circuit simulator,” *Quantum*, vol. 5, p. 497, 2021.