Noise characterisation of fault-tolerant quantum computers

Evan Timothy Hockings

A thesis submitted in fulfilment of the requirements for the degree of
Doctor of Philosophy

School of Physics
Faculty of Science
The University of Sydney

2025

Acknowledgements

There are a great many people who made this experience better than it would have been otherwise. First and foremost, I would like to thank my advisors, Andrew Doherty and Robin Harper, for deftly guiding me through this process. I often ask myself 'What would Andrew say?' and should probably do so more often! And Robin, my first port of call, answered my many questions, for which I am grateful. I would also like to thank my first advisor, Isaac Kim, for his continuing guidance even after the pandemic forced him to leave Sydney. Thanks also go to Steven Flammia for good advice over the years, and Stephen Bartlett for his support. Of course, I would like to thank my parents, Constance and Greg, for their support throughout this journey. And I would like to thank the people in the Sydney Quantum Theory group, particularly those with whom I shared an office, for making this an uncommonly social experience. Lastly, I would like to thank my friends, old and new, whose names I will not attempt to list. Know that if you could imagine your name here, it would be on the list. You are all appreciated.

The existence of this thesis feels like a minor miracle. When I started my PhD in 2021, with the COVID-19 pandemic still ongoing, I was obsessed with AI. I spent far more time reading about AI than I did reading about quantum computing! I think this thesis suffered for that choice, but it is not one I regret.

In the intervening years, AI transformed from a story to *the* story. While I am glad to have stayed the course, now, in 2025, it is time for my course to turn. I can only hope that AI is not nearly so important as I fear, and that one day I will find my way back here.

This research was supported by an Australian Government Research Training Program Scholarship. It was also supported by the Australian Research Council Centre of Excellence for Engineered Quantum Systems (CE170100009) and the U.S. Army Research Office (W911NF-21-1-0001, W911NF-23-S-0004). Moreover, the Unitary Foundation supported work on the open-source Julia package QuantumACES.

Statement of Contribution

This thesis consists of an introduction, four body chapters, and a conclusion. I wrote the introduction and conclusion. The first three of the four body chapters have been adapted from three separate research papers, with minor edits for consistent formatting. I wrote the last body chapter which describes ongoing and unpublished work. To the best of my knowledge, none of the written content in this thesis was produced by AI.

My contribution to each body chapter is described below.

Chapter 2: Scalable noise characterisation of syndrome extraction circuits with averaged circuit eigenvalue sampling

Published in: PRX Quantum 6, 010334, 2025, arXiv:2404.06545.

Author list: Evan T. Hockings, Andrew C. Doherty, Robin Harper.

Contributions: I am the lead author of this paper. I carried out the research for this project, which was initially suggested by Robin, under the guidance of Andrew and Robin. I drafted the paper and wrote the code, and Andrew and Robin provided feedback and helped to edit and clarify the paper.

Chapter 3: Improving error suppression with noise-aware decoding

Published in: arXiv:2502.21044.

Author list: Evan T. Hockings, Andrew C. Doherty, Robin Harper.

Contributions: I am the lead author of this paper. I carried out the research for this project under the guidance of Andrew and Robin. I drafted the paper and wrote the code, and Andrew and Robin provided feedback and helped to edit the paper.

Chapter 4: QuantumACES.jl: design noise characterisation experiments for quantum computers

Published in: Journal of Open Source Software 10, 107, 7707, 2025.

Author list: Evan T. Hockings.

Contributions: I wrote the paper and the code, which can be found at the GitHub repository QuantumACES.jl.

Chapter 5: A heavy hexagon memory with noise-aware decoding

Contributor list: Evan T. Hockings, Andrew C. Doherty, Benjamin J. Brown, Robin Harper.

Contributions: I wrote this thesis chapter. I carried out the research for this project under the guidance of Andrew and Robin with the exception of the following contributions. Robin and Ben designed the improved heavy hexagon code syndrome extraction circuit. Robin submitted the experiments to IBM quantum devices with minor preprocessing, and processed the results for heavy hexagon memory experiments.

Evan Timothy Hockings
February 2025

As supervisor for the candidature upon which this thesis is based, I can confirm that the authorship attribution statements above are correct.

Andrew Charles Doherty
February 2025

Statement of Originality

This is to certify that the content of this thesis is my own work. This thesis has not been submitted for any other degree or purpose.

I certify that the intellectual content of this thesis is the product of my own work, and that all assistance received in preparing this thesis and all sources have been acknowledged. This thesis contains fewer than 80,000 words, excluding the bibliography.

Evan Timothy Hockings
February 2025

Abstract

Noise is the fundamental challenge to quantum computation. Large-scale quantum computers will overcome this challenge with fault-tolerant architectures that leverage quantum error correction. This thesis introduces and investigates a scalable noise characterisation protocol suited to this key context.

First, we introduce this protocol by building upon averaged circuit eigenvalue sampling (ACES), a framework for noise characterisation experiments that simultaneously estimates the Pauli error probabilities of all gates in a Clifford circuit and captures averaged spatial correlations between gates implemented simultaneously in the layers of the circuit. We derive a figure of merit for our noise characterisation experiments that allows us to optimise their experimental design and improve the precision to which we estimate noise given fixed experimental resources. We demonstrate the scalability and performance of our protocol through circuit-level numerical simulations of the entire noise characterisation procedure for the syndrome extraction circuit of a distance-25 surface code with over 1000 qubits. I also present the open-source Julia package QuantumACES, which provides a software implementation of this protocol.

Then we demonstrate in circuit-level numerical simulation that this protocol is practically capable of calibrating a fast correlated matching decoder, enabling noise-aware decoding. We find that noise-aware decoding increases the error suppression factor of the code, leading to reductions in the logical error rate that increase exponentially with the code distance. Our results indicate that noise characterisation experiments performed and processed in seconds will suffice to calibrate decoders for fault-tolerant superconducting quantum computers.

Finally, we present several results obtained from experimentally implementing this protocol. We use the noise characterisation results to design an improved syndrome extraction circuit for a heavy hexagon memory that is adapted to the noise characteristics of the quantum device. The model error of our noise characterisation results suggests that the circuit-level Pauli noise model estimated by ACES can describe the essential features of quantum noise. We also operate a heavy hexagon memory and use noise estimates to predict performance and inform decoding.

These results demonstrate the practicality and utility of Pauli noise characterisation in the context of fault-tolerant quantum computation. Pauli noise estimates can calibrate decoders to enable noise-aware decoding and inform the co-design of quantum error correcting codes, decoders, fault-tolerant circuits, and quantum devices.

Contents

Chapter 1

Introduction

> Think of a deer skipping gracefully across a complex woodland terrain. Then think of a mathematician in an analogous situation, attempting to traverse a field of abstract algebra—stiff and achy; unnatural; almost entirely blind, able to see scarcely two yards ahead; slow and unsteady; frowning with concentration; clutching the rollator of formal proof... Maybe that happens to be the only way we can currently do it, rather than the way it really should be done.
>
> In all of this, there is room for improvement.
>
> — Nick Bostrom, *Deep Utopia: Life and Meaning in a Solved World*

1.1 Overview

Classical computers have transformed science. The maturity of classical computing architectures obscures the physics underlying classical computation beneath many layers of abstraction. In the nascent field of quantum computing, quantum physics instead demands attention and sits determinedly in the foreground. The problem is *noise*: quantum systems are delicate, and interactions with the environment can destroy quantum information, preventing us from performing useful quantum computations. Noise is also a key problem with analogue classical computers, which compute with continuous physical quantities analogous to the desired computation. Modern classical computers are digital and represent quantities discretely. This naturally performs a sort of error correction, as minor fluctuations in physical quantities leave the discrete values they represent unchanged. Discrete digital information can also be protected by classical *error correcting codes*, which detect and correct errors, though this is generally unnecessary in modern classical computers as their components are extremely reliable.

Although quantum computers are similar to analogue computers in that quantum states are a continuous representation of data, they can like digital computers use *quantum* error correcting codes to protect quantum information from noise. The power of quantum error correction lies in the fact that the ability to correct a discrete set of errors entails the ability to correct the continuous set of linear combinations of those errors [1, 2]. Quantum error correction also enables *fault-tolerant quantum computation*: various *threshold theorems* state that arbitrarily accurate quantum computation is possible

with relatively low overhead as long as the physical error rates in the components of the quantum computer are below some threshold [3–7].

The scientific and engineering challenge of designing and manufacturing quantum devices capable of operating quantum error correcting codes with logical error rates comparable to modern classical computers is seeing rapid and ongoing progress [8–19]. The substantial investment in overcoming this challenge is driven by the promise of quantum algorithms that leverage quantum mechanics to solve problems faster than the best known classical algorithms. Some algorithms of particular interest include Shor's algorithm for integer factorisation [20, 21], the HHL algorithm for sampling from the solution to a linear system of equations [22], Grover's algorithm for search [23], and methods for simulating quantum systems [24, 25]. But quantum computers are also of scientific interest, as they will allow us to probe quantum mechanics and reality in novel ways. Consider the idiosyncratic example of the simulation hypothesis, the idea that our experience of the world is constructed by a computer simulation, notably argued for by Nick Bostrom [26]. Performing and verifying the outputs of large-scale quantum computations would strongly suggest that if indeed our reality is simulated, the computer performing the simulation is not classical and leverages physical principles at least as powerful as quantum mechanics to perform computation.

Noise is the fundamental challenge to realising the promise of quantum computing. This challenge will be overcome through fault-tolerant quantum computation [27–29], which encodes quantum information in quantum error correcting codes and manipulates this information in ways protected against errors. Replacing physical state preparations, gates, and measurements with encoded logical equivalents entails regularly measuring the parity checks of quantum error correcting codes to enable the detection and correction of errors. In a typical fault-tolerant architecture, the syndrome extraction circuits that measure these parity checks represent the bulk of the physical qubits and gate operations.

I believe that noise characterisation of quantum devices should focus on the key context of quantum error correction and fault tolerance. Highly scalable noise characterisation protocols are necessary in this context. Specifically, I believe syndrome extraction circuits are the key target of noise characterisation experiments, and are representative of operations in a fault-tolerant quantum computer. Noise estimates can calibrate decoders of quantum error correcting codes to improve their performance. They can also inform the co-design of quantum error correcting codes, decoders, fault-tolerant circuits, and quantum devices.

Accordingly, Chapter 2 introduces a scalable Pauli noise characterisation protocol suited to the context of quantum error correction and, in particular, syndrome extraction circuits. We derive a figure of merit for the accuracy of noise characterisation experiments and use it to optimise their experimental design, improving the precision to which we estimate noise given fixed experimental resources. Through large-scale numerical simulations of surface codes, a well-studied family of quantum error correcting codes, we demonstrate that our protocol is practically capable of characterising noise at the scales required for quantum error correction.

Armed with this protocol, Chapter 3 demonstrates that the noise estimates can be used to calibrate the decoders of quantum error correcting codes to improve their performance. This improves the error suppression factor of the code, yielding reductions in the logical error rate that increase exponentially with the code distance. These results demonstrate the practicality and utility of Pauli noise characterisation for noise-aware decoding in the context of quantum error correction at scale.

Following this, Chapter 4 describes the open-source Julia package QuantumACES which produces the results described in Chapter 2 and Chapter 3 merely on a laptop. I developed QuantumACES with the aim of making a state-of-the-art scalable Pauli noise characterisation protocol openly accessible. Benchmarking quantum hardware, particularly in the key context of quantum error correction, should be free and easy.

Then Chapter 5 describes ongoing work to experimentally implement the techniques described in Chapter 2 and Chapter 3, with the help of the code described in Chapter 4. We practically implement our noise characterisation protocol on IBM quantum devices and examine the model error associated with our experimental noise estimates. Then we use these noise estimates to inform the design of syndrome extraction circuits for a particular quantum error correcting code called the heavy hexagon code, and also use them to calibrate the decoder prior.

Finally, Chapter 6 concludes with a summary of the results in this thesis. I also offer a brief perspective on the future of noise characterisation in fault-tolerant quantum computation.

While Chapter 2, Chapter 3, and Chapter 4 can be read as self-contained works, Chapter 5 builds upon the results described in these previous chapters. These chapters rely on some key underlying concepts introduced in the remainder of this chapter. This introduction endeavours to be precise while using only relatively elementary mathematical formalism. First Section 1.2 introduces concepts and formalism used in quantum information, particularly in the context of noise characterisation, including the density operator formalism in Section 1.2.1, the Pauli and Clifford groups in Section 1.2.2, and Pauli channels in Section 1.2.3. Then Section 1.3 introduces quantum error correction, first presenting some results about symplectic vector spaces in Section 1.3.1 to introduce stabiliser codes in Section 1.3.2. Then Section 1.3.3 introduces topological codes, a class of quantum error correcting codes with helpful properties for practical implementation, before Section 1.3.4 closes with a discussion of fault-tolerant quantum circuits.

1.2 Quantum information

This section introduces formalism and concepts used in quantum information. This introduction aims to give a flavour for some core concepts underpinning the ideas in this thesis while avoiding complicated mathematical formalism.

1.2.1 Density operators and quantum operations

First we introduce mathematical formalism used to describe quantum systems, drawing on [1]. Closed quantum systems are usually described with the *state vector* formalism. This formalism describes the state of a quantum system by a state vector $|\psi\rangle$ in the state space, a d-dimensional Hilbert space $\mathcal{H} \cong \mathbb{C}^d$ which is a (complete) complex inner product space equipped with the usual ℓ_2 inner product. In the context of quantum computation, we are particularly interested in *qubits*, the quantum mechanical analogue of classical bits. Qubits are described by a two-dimensional Hilbert space $\mathbb{C}^2$ spanned by the orthonormal *computational basis* states $|0\rangle$ and $|1\rangle$. Then, since the state space of a composite quantum system is the tensor product of the state spaces of its components, an n-qubit system is described by a 2^n-dimensional Hilbert space $\mathcal{H} \cong \mathbb{C}^{2^n}$.

As this thesis is particularly interested in the effects of noise on quantum systems, we therefore consider open quantum systems which are more conveniently described with

the *density operator* formalism, which is mathematically equivalent to the state vector formalism. A density operator ρ is a positive semi-definite unit trace operator on the state space which lies in $\mathbb{C}^{d\times d}$ and can be written as

$$\rho = \sum_i p_i |\psi_i\rangle\langle\psi_i|, \tag{1.1}$$

a probabilistic mixture of state vectors $|\psi_i\rangle$ with non-negative probabilities p_i that sum to one.

The evolution of a closed quantum system is described in the state vector formalism by a unitary operator U on $\mathcal{H}$ which acts as $|\psi'\rangle = U|\psi\rangle$. In the density operator formalism, it instead acts as

$$\rho' = U\rho U^\dagger. \tag{1.2}$$

Measurement is described by a collection of measurement operators $\{M_m\}$ indexed by the measurement outcomes m which satisfy the *completeness relation*

$$\sum_m M_m^\dagger M_m = I. \tag{1.3}$$

In the state vector formalism, the measurement outcome probabilities are given by $p(m) = \langle\psi|M_m^\dagger M_m|\psi\rangle$, and the state after the measurement outcome m is $|\psi_m\rangle = M_m|\psi\rangle/\sqrt{p(m)}$. In the density operator formalism, the measurement outcome probabilities are instead given by

$$p(m) = \operatorname{tr}(M_m^\dagger M_m \rho), \tag{1.4}$$

and the state after the measurement outcome m is

$$\rho_m = \frac{M_m \rho M_m^\dagger}{p(m)}. \tag{1.5}$$

We will work with the more convenient *projective* measurement formalism, which is in fact equivalent to the general measurement formalism with unitary evolution. In a projective measurement, the measurement operators M_m are orthogonal projections, that is, Hermitian with $M_m M_{m'} = \delta_{mm'} M_m$. Then these measurement operators correspond to the spectral decomposition of a measurement *observable* M given by

$$M = \sum_m m M_m. \tag{1.6}$$

Moreover, the expectation of the observable M is simply $\langle\psi|M|\psi\rangle$ or $\operatorname{tr}(M\rho)$ in the state vector and density operator formalisms, respectively.

The formalism of quantum operations can be used to describe the evolution of open quantum systems. A quantum operation is a *completely positive* and *non-trace-increasing* linear map on the space of density operators $\mathcal{L}\colon \mathbb{C}^{d\times d} \to \mathbb{C}^{d\times d}$. They are capable of representing unitary evolution

$$\rho' = \mathcal{L}(\rho) = U\rho U^\dagger, \tag{1.7}$$

and measurement

$$\rho_m = \mathcal{L}_m(\rho) = M_m \rho M_m^\dagger. \tag{1.8}$$

The *Kraus representation* characterises quantum operations as

$$\mathcal{L}(\rho) = \sum_{k=1}^{K} L_k \rho L_k^\dagger, \tag{1.9}$$

where the *Kraus operators* $\{L_k\}$, of which there at most $K \leq d^2$, satisfy the completeness relation

$$\sum_{k=1}^{K} L_k^\dagger L_k \leq I. \tag{1.10}$$

We will focus on *quantum channels*, which are trace-preserving quantum operations often referred to as *completely positive trace-preserving* (CPTP) maps, for which this completeness relation inequality is saturated.

In this thesis, we will describe noise with quantum channels. This reduces the problem of noise characterisation to one of *quantum process tomography*, the problem of estimating a quantum channel $\mathcal{L}$. However, quantum process tomography generally requires estimating $\mathcal{O}(d^4)$ parameters, and since $d = 2^n$ for n-qubit systems, this quickly becomes infeasible. We therefore seek to model noise processes with less general representations that have fewer parameters in order to tractably characterise noise.

1.2.2 Pauli and Clifford groups

Next we introduce the Pauli and Clifford groups, which are commonly encountered in quantum information, drawing on [30]. The single-qubit Pauli operators are defined as

$$I = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}, \quad X = \begin{bmatrix} 0 & 1 \\ 1 & 0 \end{bmatrix}, \quad Z = \begin{bmatrix} 1 & 0 \\ 0 & -1 \end{bmatrix}, \quad Y = \begin{bmatrix} 0 & -i \\ i & 0 \end{bmatrix}. \tag{1.11}$$

These have a number of convenient properties. They are Hermitian and unitary, and therefore involutions, meaning that they square to the identity. They also span the space $\mathbb{C}^{2\times 2}$ of 2×2 complex matrices. Moreover, the non-identity Pauli operators have two eigenvalues, $+1$ and -1, and their eigenvectors and spectral decompositions are conventionally denoted

$$X = |+\rangle\langle +| - |-\rangle\langle -|, \quad Z = |0\rangle\langle 0| - |1\rangle\langle 1|, \quad Y = i|0\rangle\langle 1| - i|1\rangle\langle 0|. \tag{1.12}$$

The n-qubit Pauli operators are n-fold tensor products of single-qubit Pauli operators, so that there are 4^n n-qubit Pauli operators, which also share these properties. In particular, the n-qubit Pauli operators span the space $\mathbb{C}^{d\times d}$ of $d \times d$ complex matrices, where we take $d = 2^n$. The natural inner product on this space is the *trace* or *Hilbert-Schmidt* inner product

$$\langle A, B\rangle = \operatorname{tr}\left(A^\dagger B\right). \tag{1.13}$$

The Pauli operators are orthogonal with respect to this inner product with norm $\sqrt{d}$, and so the Pauli operators normalised by $\sqrt{d}$ form an orthonormal basis for $\mathbb{C}^{d\times d}$. Indeed, this fact is particularly important in the context of quantum error correction, as it means that any error can be expressed as a linear combination of Pauli errors. It therefore suffices to focus on the correction of Pauli errors when developing quantum error correcting codes.

The Pauli operators form a group under multiplication called the *Pauli group* $\mathbf{P}^n$. We can index the n-qubit Pauli operators by length $2n$ bit strings

$$\boldsymbol{a} = (\boldsymbol{a}^{(x)}, \boldsymbol{a}^{(z)}) = (a_1^{(x)}, \ldots, a_n^{(x)}, a_1^{(z)}, \ldots, a_n^{(z)}), \tag{1.14}$$

where $a_j^{(x)}$ and $a_j^{(z)}$ refer to whether Pauli X and Pauli Z are present on the jth qubit according to the equation

$$P_{\boldsymbol{a}} = \bigotimes_{j=1}^{n} i^{a_j^{(x)} a_j^{(z)}} X^{a_j^{(x)}} Z^{a_j^{(z)}}. \tag{1.15}$$

Then the Pauli group contains all 4^n Pauli operators indexed by bit strings in $\mathbb{Z}_2^{2n}$ described by this equation, which ensures that they are Hermitian, as well as phase factors $\langle i \rangle = \{\pm 1, \pm i\}$. Conventionally, if we refer to a bit string $\boldsymbol{a} \in \mathbf{P}^n$ we mean the unit phase element given by Equation 1.15. We can also define the *support* of a Pauli $\boldsymbol{a} \in \mathbf{P}^n$ to be the set of qubits on which it is not the identity, that is

$$\operatorname{supp}(\boldsymbol{a}) = \{j \in [n] \colon \{a_j^{(x)}, a_j^{(z)}\} \neq \{0, 0\}\}. \tag{1.16}$$

The *weight* of a Pauli $|\boldsymbol{a}| = |\operatorname{supp}(\boldsymbol{a})|$ is simply the size of its support.

Now consider the Pauli quotient group $\mathrm{P}^n = \mathbf{P}^n / \langle i \rangle$. We have quotiented the Pauli group by the phase factors, which are the centre of the group as they commute with all elements, so P^n can be thought of as the Abelianisation of $\mathbf{P}^n$. Then, taking elements $P_{\boldsymbol{a}}$ to refer to the unit phase coset representatives in P^n, we have that

$$P_{\boldsymbol{a}} P_{\boldsymbol{b}} = P_{\boldsymbol{a}+\boldsymbol{b}}. \tag{1.17}$$

This defines an isomorphism $\mathrm{P}^n \cong \mathbb{Z}_2^{2n}$, where the group operation of matrix multiplication in P^n is equivalent to vector addition in $\mathbb{Z}_2^{2n}$. We will therefore refer to bit strings $\boldsymbol{a} \in \mathrm{P}^n$, and call P^n the bit string vector space.

Separately, we can define the commutation relation of two Pauli operators with a *symplectic bilinear form* $\omega \colon \mathrm{P}^n \times \mathrm{P}^n \to \mathbb{Z}_2$, which is also symmetric as we work over $\mathbb{Z}_2$, defined as

$$\omega(\boldsymbol{a}, \boldsymbol{b}) = \boldsymbol{a}^{(x)} \cdot \boldsymbol{b}^{(z)} + \boldsymbol{a}^{(z)} \cdot \boldsymbol{b}^{(x)}, \tag{1.18}$$

such that

$$P_{\boldsymbol{a}} P_{\boldsymbol{b}} = (-1)^{\omega(\boldsymbol{a},\boldsymbol{b})} P_{\boldsymbol{b}} P_{\boldsymbol{a}}. \tag{1.19}$$

A symplectic bilinear form is linear in both arguments, *alternating*, so that $\omega(\boldsymbol{a}, \boldsymbol{a}) = 0$ for all $\boldsymbol{a}$, and *non-degenerate*, so that $\omega(\boldsymbol{a}, \boldsymbol{b}) = 0$ for all $\boldsymbol{b}$ implies $\boldsymbol{a} = \boldsymbol{0}$. We equip the bit string vector space P^n with this form such that it is a $2n$-dimensional symplectic vector space.

This allows us to define a *stabiliser group* as a strict subspace $S \subset \mathrm{P}^n$ such that $\omega(\boldsymbol{a}, \boldsymbol{b}) = 0$ for all $\boldsymbol{a}, \boldsymbol{b} \in S$, that is, a subspace of mutually commuting Paulis. Stabiliser groups are usually defined as a mutually commuting subgroup of $\mathbf{P}^n$ alongside the phase condition $-I \notin S$, as it is often important to track the phases of the stabiliser group elements. For simplicity, we will avoid treating global phases of Pauli group elements. However, note that this phase condition implies that stabiliser groups do not contain anti-Hermitian Pauli group elements with phase $\pm i$, and also do not contain both a Pauli group element and its negative.

Then a state $|\psi\rangle$ is *stabilised* by S if $P_{\boldsymbol{a}}|\psi\rangle = |\psi\rangle$ for all $P_{\boldsymbol{a}} \in S$. We will later show that a *maximal* stabiliser group is n-dimensional and therefore contains 2^n elements. In fact, we can uniquely specify a *stabiliser state* $|\psi\rangle$ by the maximal stabiliser group by which it is stabilised.

The *Clifford group* is sometimes defined as the group of unitary operators that normalise the Pauli group, that is, unitary operators U such that for any Pauli $P_{\boldsymbol{a}}$, there exists some Pauli $P_{\boldsymbol{b}}$ such that $UP_{\boldsymbol{a}}U^\dagger = P_{\boldsymbol{b}}$. However, the centre of this group is infinite and consists of all phases $e^{i\theta}I$. It is more convenient to define the Clifford group $\mathbf{C}^n$ as the group generated by the Hadamard, phase, and controlled-X gates, written H_j, S_j, and $C_i(X_j)$, for control qubits i and target qubits $j \neq i$. The matrix representations of these gates are

$$H_1 = \frac{1}{\sqrt{2}}\begin{bmatrix}1 & 1\\ 1 & -1\end{bmatrix}, \quad S_1 = \begin{bmatrix}1 & 0\\ 0 & i\end{bmatrix}, \quad C_1(X_2) = \begin{bmatrix}1 & 0 & 0 & 0\\ 0 & 1 & 0 & 0\\ 0 & 0 & 0 & 1\\ 0 & 0 & 1 & 0\end{bmatrix}, \tag{1.20}$$

and another common gate is the controlled-Z gate $C_i(Z_j) = H_j C_i(X_j) H_j$, given by

$$C_1(Z_2) = \begin{bmatrix}1 & 0 & 0 & 0\\ 0 & 1 & 0 & 0\\ 0 & 0 & 1 & 0\\ 0 & 0 & 0 & -1\end{bmatrix}. \tag{1.21}$$

Defining the Clifford group this way yields phases $\langle\eta\rangle$ for $\eta = (1+i)/\sqrt{2}$, where we note that $\eta^2 = i$.

It is also convenient to consider the Clifford quotient group $\mathrm{C}^n = \mathbf{C}^n/\langle\eta, \mathbf{P}^n\rangle$, where we quotient by the Pauli group and phase factors. This group has a number of elements

$$|\mathrm{C}^n| = 2^{n^2}\prod_{j=1}^{n} 4^j - 1. \tag{1.22}$$

This Clifford quotient group is isomorphic to the symplectic group $\mathrm{Sp}(2n, \mathbb{Z}_2)$ and consists of all linear transformations on $\mathbb{Z}_2^{2n} \cong \mathrm{P}^n$ that preserve the symplectic form ω. Specifically, this means that $\omega(M\boldsymbol{a}, M\boldsymbol{b}) = \omega(\boldsymbol{a}, \boldsymbol{b})$ for all $\boldsymbol{a}, \boldsymbol{b} \in \mathrm{P}^n$ and $M \in \mathrm{Sp}(2n, \mathbb{Z}_2)$, making this a *symplectic representation* of the Clifford group.

The symplectic representation of the action of the Clifford group on the Pauli group enables the efficient classical simulation of *stabiliser circuits*. This simulation is performed by tracking the effect of Clifford gates, as well as computational basis measurements, on the stabilisers of a stabiliser state, enabling n-qubit stabiliser circuits to be simulated with polynomially many resources as opposed to the exponentially many resources required for general quantum circuits. This fact is known as the *Gottesman-Knill theorem* [31, 32]. Indeed, augmenting stabiliser circuits with only the gate $T = \sqrt{S}$ yields *universal quantum computation*, enabling the approximation of any unitary operator to arbitrary precision with an efficiency described by the *Solovay-Kitaev theorem* [1, 33, 34].

Stabiliser circuits are common in quantum error correction and, in particular, the syndrome extraction circuits of quantum error correcting codes are stabiliser circuits. Thus noise characterisation efforts can safely focus primarily on characterising noise in stabiliser circuits, leveraging the ability to efficiently simulate their action classically, as we will do in this thesis.

1.2.3 Pauli channels

Now we are ready to introduce the main tool we will use to model noise in quantum systems, namely Pauli channels, drawing on [35]. A *Pauli channel* is a quantum channel

that can be written in the form

$$\mathcal{E}(\rho) = \sum_{\boldsymbol{a}\in \mathrm{P}^n} p_{\boldsymbol{a}} P_{\boldsymbol{a}} \rho P_{\boldsymbol{a}}, \tag{1.23}$$

where the *Pauli error probabilities* $p_{\boldsymbol{a}}$ are non-negative and sum to one, forming a probability distribution over Pauli errors. Pauli channels represent quantum noise as a probabilistic mixture of Pauli errors, governed by these Pauli error probabilities. The problem of learning a Pauli channel reduces to estimating the 4^n Pauli error probabilities, where there are $4^n - 1$ free parameters. This is more tractable than quantum process tomography, though it still requires estimating exponentially many parameters.

Although the Pauli operators are not valid density operators, nevertheless consider the action of a Pauli channel on a Pauli operator

$$\mathcal{E}(P_{\boldsymbol{b}}) = \sum_{\boldsymbol{a}\in \mathrm{P}^n} p_{\boldsymbol{a}} P_{\boldsymbol{a}} P_{\boldsymbol{b}} P_{\boldsymbol{a}} = \left(\sum_{\boldsymbol{a}\in \mathrm{P}^n} (-1)^{\omega(\boldsymbol{a},\boldsymbol{b})} p_{\boldsymbol{a}} \right) P_{\boldsymbol{b}} = \lambda_{\boldsymbol{b}} P_{\boldsymbol{b}}. \tag{1.24}$$

Thus we see that the Pauli operators are the eigenvectors of the Pauli channel with *Pauli channel eigenvalues* $\lambda_{\boldsymbol{b}}$. Indeed, the Pauli error probabilities and Pauli channel eigenvalues are related by the *Walsh-Hadamard transform* ordered by the symplectic form ω, such that in reverse we have

$$p_{\boldsymbol{a}} = \frac{1}{4^n} \sum_{\boldsymbol{b}\in \mathrm{P}^n} (-1)^{\omega(\boldsymbol{a},\boldsymbol{b})} \lambda_{\boldsymbol{b}}. \tag{1.25}$$

As the Walsh-Hadamard transform is a discrete Fourier transform, we can think of the error probabilities and eigenvalues as Fourier duals of each other. This allows us to learn the Pauli channel eigenvalues instead of the Pauli error probabilities.

Specifically, consider the eigenbasis of $|\psi_{\boldsymbol{s}}^{\boldsymbol{a}}\rangle$ of $P_{\boldsymbol{a}}$ consisting of all 2^n sign configurations of tensor products of single-qubit Pauli eigenstates indexed by the length n bit string $\boldsymbol{s}$. Then $|\psi_{\boldsymbol{s}}^{\boldsymbol{a}}\rangle$ is an eigenstate of $P_{\boldsymbol{a}}$ with eigenvalue $(-1)^s$, where s is the parity of $\boldsymbol{s}$. Now suppose we prepare eigenstates of the Pauli operator $P_{\boldsymbol{a}}$ uniformly at random, apply the channel $\mathcal{E}$ some number of times m, and then measure the expectation value of the Pauli operator $P_{\boldsymbol{a}}$. Then, referencing Equation 1.24 and calculating, we see that

$$\begin{aligned}
\frac{1}{2^n} \sum_{\boldsymbol{s}} (-1)^s \operatorname{tr}\left(P_{\boldsymbol{a}} \mathcal{E}^m(|\psi_{\boldsymbol{s}}^{\boldsymbol{a}}\rangle\langle\psi_{\boldsymbol{s}}^{\boldsymbol{a}}|)\right) &= \frac{1}{2^n} \operatorname{tr}\left(P_{\boldsymbol{a}} \mathcal{E}^m(P_{\boldsymbol{a}})\right) && (1.26)\\
&= \frac{1}{2^n} \lambda_{\boldsymbol{a}}^m \operatorname{tr}\left(P_{\boldsymbol{a}} P_{\boldsymbol{a}}\right) && (1.27)\\
&= \lambda_{\boldsymbol{a}}^m. && (1.28)
\end{aligned}$$

Up to some sign corrections, the measurement outcomes directly estimate the Pauli channel eigenvalue $\lambda_{\boldsymbol{a}}$ raised to the power m. This is the fundamental strategy underlying Pauli channel estimation techniques.

Define the *Pauli twirl* of a quantum channel $\mathcal{L}$ as the average over the Pauli group $\mathbf{P}^n$, namely

$$\mathcal{L}^{\mathrm{P}^n}(\rho) = \frac{1}{4^n} \sum_{\boldsymbol{a}\in \mathrm{P}^n} \sum_k \left(P_{\boldsymbol{a}} L_k P_{\boldsymbol{a}}^\dagger\right) \rho \left(P_{\boldsymbol{a}} L_k P_{\boldsymbol{a}}^\dagger\right)^\dagger. \tag{1.29}$$

As the Pauli operators form an orthogonal Hermitian basis, we can express the operation elements L_k in the form

$$L_k = \frac{1}{2^n} \sum_{\boldsymbol{b} \in \mathrm{P}^n} \mathrm{tr}\left(P_{\boldsymbol{b}} L_k\right) P_{\boldsymbol{b}} = \sum_{\boldsymbol{b} \in \mathrm{P}^n} l_{k\boldsymbol{b}} P_{\boldsymbol{b}}, \tag{1.30}$$

where the coefficients $l_{k\boldsymbol{b}}$ are real owing to the Hermiticity of the Pauli operators. Then, calculating, we find that

$$\mathcal{L}^{\mathrm{P}^n}(\rho) = \frac{1}{4^n} \sum_{\boldsymbol{a},\boldsymbol{b},\boldsymbol{c} \in \mathrm{P}^n} \left(\sum_k l_{k\boldsymbol{b}} l_{k\boldsymbol{c}} \right) \left(P_{\boldsymbol{a}} P_{\boldsymbol{b}} P_{\boldsymbol{a}}\right) \rho \left(P_{\boldsymbol{a}} P_{\boldsymbol{c}} P_{\boldsymbol{a}}\right) \tag{1.31}$$

$$= \frac{1}{4^n} \sum_{\boldsymbol{a},\boldsymbol{b},\boldsymbol{c} \in \mathrm{P}^n} \left(\sum_k l_{k\boldsymbol{b}} l_{k\boldsymbol{c}} \right) (-1)^{\omega(\boldsymbol{a},\boldsymbol{b}+\boldsymbol{c})} P_{\boldsymbol{b}} \rho P_{\boldsymbol{c}}. \tag{1.32}$$

The non-degeneracy of ω implies that for $\boldsymbol{a} \neq \boldsymbol{0}$ the linear functional $f_{\boldsymbol{a}}(\boldsymbol{b}) = \omega(\boldsymbol{a}, \boldsymbol{b})$ has kernel dimension $2n-1$ by the rank-nullity theorem. This means 2^{2n-1} elements of the Pauli group commute with $\boldsymbol{a} \neq \boldsymbol{0}$, and therefore the other 2^{2n-1} elements must anticommute, whereas all elements commute with $\boldsymbol{0}$. That is, half of the Pauli group elements commute and half anticommute with any non-zero Pauli, a fact that is stated without proof as Lemma 1 in [35]. This leads to the observation

$$\frac{1}{4^n} \sum_{\boldsymbol{a} \in \mathrm{P}^n} (-1)^{\omega(\boldsymbol{a},\boldsymbol{b})} = \delta_{\boldsymbol{b}\boldsymbol{0}}. \tag{1.33}$$

Now noting that $\boldsymbol{b} + \boldsymbol{c} = \boldsymbol{0}$ if and only if $\boldsymbol{b} = \boldsymbol{c}$ since we are working over $\mathbb{Z}_2$, substituting yields that

$$\mathcal{L}^{\mathrm{P}^n}(\rho) = \sum_{\boldsymbol{b},\boldsymbol{c} \in \mathrm{P}^n} \left(\sum_k l_{k\boldsymbol{b}} l_{k\boldsymbol{c}} \right) \delta_{\boldsymbol{b}\boldsymbol{c}} P_{\boldsymbol{b}} \rho P_{\boldsymbol{c}} \tag{1.34}$$

$$= \sum_{\boldsymbol{b} \in \mathrm{P}^n} \left(\sum_k l_{k\boldsymbol{b}}^2 \right) P_{\boldsymbol{b}} \rho P_{\boldsymbol{b}}. \tag{1.35}$$

This is a Pauli channel with Pauli error probabilities

$$p_{\boldsymbol{b}} = \sum_k l_{k\boldsymbol{b}}^2. \tag{1.36}$$

This fact, namely that the Pauli twirl of a quantum channel is a Pauli channel, is the key result that motivates characterising noise as Pauli noise. Specifically, if we operate quantum circuits interspersed with randomly chosen Pauli gates, then the resulting noise is Pauli noise. This is known as Pauli frame randomisation in the context of stabiliser circuits [36], where the sign correction resulting from these Pauli gates can be efficiently computed classically. In arbitrary quantum circuits, this idea is instead realised with the randomised compiling protocol [37]. Moreover, the process of quantum error correction itself tailors noise into Pauli noise [38–40]. Therefore, we focus in this thesis on learning Pauli noise, specifically by estimating the eigenvalues of Pauli channels.

It is generally intractable to learn a Pauli channel on n qubits, as there are $4^n - 1$ non-identity eigenvalues that must be estimated. Instead, we can consider a *circuit-level*

noise model which associates a noise channel with each gate in the circuit. This can be thought of as a local noise model where the locality is defined by the circuit, as has the cost of neglecting correlated errors. However, it also drastically reduces the number of parameters that need to be estimated, and learning a circuit-level Pauli noise model is highly tractable, as we will see in Chapter 2.

1.3 Quantum error correction

This section introduces concepts and formalism used in quantum error correction through the lens of stabiliser codes. Although there are other ways to construct quantum error correcting codes, the stabiliser code formalism is powerful, general, and allows us to describe the quantum error correcting codes we will consider in this thesis. We will introduce stabiliser codes in a somewhat idiosyncratic manner, in particular by first sketching results about symplectic vector spaces.

To see why this is a natural approach, let us first build upon the notation in Section 1.2.2. Define the *centraliser* of a subspace S of P^n as the subspace $C(S)$ of elements that commute with those in S, namely

$$C(S) = \{\boldsymbol{a} \in \mathrm{P}^n \colon \forall \boldsymbol{s} \in S, \omega(\boldsymbol{a}, \boldsymbol{s}) = 0\}. \tag{1.37}$$

The centraliser is also called the commutant and, as Paulis only commute or anticommute, the centraliser coincides with the normaliser of S, the subspace of elements that leave S fixed under conjugation. While this concept can be defined identically for subsets of P^n in the same way as for subspaces, that will not be necessary here.

In the language of symplectic vector spaces, the centraliser $C(S)$ is the symplectic complement of S, and stabiliser groups are isotropic, meaning that $S \subseteq C(S)$. Moreover, we will see that the maximum dimension of an isotropic subspace is n, and an n-dimensional isotropic subspace is Lagrangian, meaning that $S = C(S)$. Thus we will elucidate key properties of stabiliser codes by first introducing results about symplectic vector spaces.

1.3.1 Symplectic vector spaces

In this section, we introduce a few results about symplectic vector spaces, drawing on [41], which we will then apply to characterise stabiliser codes. While we will present proofs of these results that assume some familiarity with linear and abstract algebra, an understanding of the proofs is inessential and they can safely be skipped.

Let V be a $2n$-dimensional vector space over the field F, and let $\omega \colon V \times V \to F$ be an alternating and non-degenerate symplectic bilinear form on V so that (V, ω) is a symplectic vector space. Recall that ω is alternating if $\omega(v, v) = 0$ for all $v \in V$, and since $\omega(v, w) + \omega(w, v) = 0$ by linearity, this implies skew-symmetry $\omega(v, w) = -\omega(w, v)$. Also recall that ω is non-degenerate if $\omega(v, w) = 0$ for all $w \in V$ implies that $v = 0$. The *symplectic complement* of a subspace $W \subseteq V$ is

$$W^\omega = \{v \in V \colon \forall w \in W, \omega(v, w) = 0\}. \tag{1.38}$$

Then W is *isotropic* if $W \subseteq W^\omega$, *coisotropic* if $W^\omega \subseteq W$, and *Lagrangian* if $W = W^\omega$.

Next define the *dual map* $\phi \colon V \to V^*$ which maps elements of V to linear functionals in the dual space V^* as $\phi(v) = \phi_v$ that act as $\phi_v(w) = \omega(w, v)$ for all $w \in V$. The dual

map ϕ is linear by the linearity of ω, and since ω is non-degenerate, its kernel is trivial, so it is injective. Since V is finite-dimensional, it has the same dimension as its dual V^*, so ϕ is surjective and hence an isomorphism. Now for any subspace $W \subseteq V$, consider the map $\phi^{(W)}\colon V \to W^*$ whose linear functionals similarly act as $\phi_v^{(W)}(w) = \omega(w, v)$ for all $w \in W$. Clearly $\phi^{(W)}$ is surjective with kernel W^ω, and since W and W^* have the same dimension, the rank-nullity theorem yields

$$\dim W + \dim W^\omega = \dim V. \tag{1.39}$$

This implies a number of results, including that the double complement $W^{\omega\omega} = W$, so W is isotropic if and only if W^ω is coisotropic. Also, an isotropic subspace can have dimension at most n, and Lagrangian subspaces are exactly isotropic subspaces of dimension n.

Now consider a basis $\{u_1, \ldots, u_n\}$ of a Lagrangian subspace L. We will show that this can be extended with a *dual basis* $\{v_1, \ldots, v_n\}$ to obtain a *symplectic basis* for V which has the commutation properties

$$\omega(u_i, u_j) = \omega(v_i, v_j) = 0, \quad \omega(u_i, v_j) = \delta_{ij}, \quad \forall i, j \in [n], \tag{1.40}$$

where we use the notation $[n] = \{1, \ldots, n\}$. We will first construct the set $\{v_1, \ldots, v_n\}$ using a symplectic Gram-Schmidt process to ensure the commutation properties, and then show that the resulting set $\{u_1, \ldots, u_n, v_1, \ldots, v_n\}$ is linearly independent and hence, being maximal, a basis for V.

First, let us construct the set $\{v_1, \ldots, v_n\}$ by induction using a symplectic Gram-Schmidt procedure. Suppose we have obtained vectors $\{v_1, \ldots, v_{i-1}\}$ which obey the desired commutation properties, then by the non-degeneracy of ω there exists some $w \in V$ such that $\omega(w, u_i) \neq 0$. Ordinarily we would normalise by $\omega(w, u_i)$, but we work over the field $\mathbb{Z}_2$ which implies that $\omega(w, u_i) = 1$. Then define

$$v_i = w - \sum_{k=1}^{i-1} \omega(w, u_k) v_k - \sum_{k=1}^{i-1} \omega(w, v_k) u_k. \tag{1.41}$$

A simple calculation using the induction hypothesis shows that $\omega(u_j, v_i) = \delta_{ji}$, and similarly $\omega(v_i, v_j) = 0$, where in particular $\omega(v_i, v_i) = 0$ follows because ω is alternating. This completes the proof by induction.

Now consider the set $\{u_1, \ldots, u_n, v_1, \ldots, v_n\}$ and suppose

$$\sum_{i=1}^{n} a_i u_i + \sum_{i=1}^{n} b_i v_i = 0. \tag{1.42}$$

Then the commutation properties show that

$$\omega\left(u_j, \sum_{i=1}^{n} a_i u_i + \sum_{i=1}^{n} b_i v_i\right) = b_j = 0, \tag{1.43}$$

$$\omega\left(\sum_{i=1}^{n} a_i u_i + \sum_{i=1}^{n} b_i v_i, v_j\right) = a_j = 0, \tag{1.44}$$

for all $j \in [n]$. Thus the set $\{u_1, \ldots, u_n, v_1, \ldots, v_n\}$ is linearly independent, and since it has $2n$ elements, we conclude that it is the desired symplectic basis for V.

Finally, let $W \subseteq V$ be a coisotropic subspace and consider the quotient space $\bar{W} = W/W^\omega$, called the *symplectic reduction* of V by W. We will show that this is a symplectic vector space $(\bar{W}, \bar{\omega})$ whose symplectic bilinear form is inherited from ω on V. Moreover, letting $L \subseteq W$ be a Lagrangian subspace, then $\bar{L} = L/W^\omega$ is also a Lagrangian subspace of $(\bar{W}, \bar{\omega})$.

To show the former, write equivalence classes as $[w] = w + W^\omega \in \bar{W}$ for $w \in W$, then we claim $\bar{\omega}([v], [w]) = \omega(v, w)$ for $v, w \in W$ is a well-defined symplectic form on $\bar{W}$. Since $W^\omega \subseteq W$ as W is coisotropic, $\omega(v, w) = 0$ for all $v \in W^\omega$ and $w \in W$, so $\bar{\omega}$ only depends on equivalence classes and hence is well-defined. It inherits bilinearity and the alternating property from ω. Lastly, for any $w \in W$, if $\omega(v, w) = 0$ for all $v \in W$, then $w \in W^\omega$, so $\bar{\omega}$ is non-degenerate. Hence $\bar{\omega}$ is a symplectic bilinear form and $(\bar{W}, \bar{\omega})$ is a symplectic vector space.

For the latter, let $[w] \in \bar{W}$ such that $\bar{\omega}([w], [v]) = 0$ for all $[v] \in \bar{L}$, then $\omega(w, v) = 0$ for all $v \in L$, so $w \in L^\omega$. But since $L^\omega = L$, we conclude that $\bar{L}^\omega = \bar{L}$ and therefore $\bar{L}$ is a Lagrangian subspace.

These results allow us to define stabiliser codes as symplectic reductions of the Pauli group. As we have seen, these symplectic reductions behave like smaller and redundantly encoded versions of the original Pauli group.

1.3.2 Stabiliser codes

We are now ready to define stabiliser codes, drawing on [30]. For more standard introductions, see [1, 2, 31]. As previously discussed, the symplectic vector space results presented in Section 1.3.1 ignore the phase conditions usually imposed on stabiliser groups. This simplifies the presentation but does not materially change the results. For an exacting mathematical treatment, refer to [42].

A *stabiliser code* encoding k logical qubits in n physical qubits is defined by a generating set $\{\boldsymbol{s}_1, \ldots, \boldsymbol{s}_n\}$ for a maximal stabiliser group, which we divide into two sets. The $n-k$ *stabiliser generators* generate the stabiliser group $S = \langle \boldsymbol{s}_1, \ldots, \boldsymbol{s}_{n-k} \rangle$, and the k *logical stabiliser generators* generate the logical stabiliser group $L_S = \langle \boldsymbol{s}_{n-k+1}, \ldots, \boldsymbol{s}_n \rangle$. We extend this to a symplectic basis for the Pauli group with the dual basis $\{\boldsymbol{r}_1, \ldots, \boldsymbol{r}_n\}$, which we again divide into two sets. The $n-k$ *destabiliser generators* generate the destabiliser group $R = \langle \boldsymbol{r}_1, \ldots, \boldsymbol{r}_{n-k} \rangle$, and the k *logical destabiliser generators* generate the logical destabiliser group $L_R = \langle \boldsymbol{r}_{n-k+1}, \ldots, \boldsymbol{r}_n \rangle$. Then the *logical group* of the code is defined as $L = \langle \boldsymbol{s}_{n-k+1}, \ldots, \boldsymbol{s}_n, \boldsymbol{r}_{n-k+1}, \ldots, \boldsymbol{r}_n \rangle = L_S \oplus L_R$. Observe that the logical group elements all commute with S and $C(S) = S \oplus L$. We can also use this symplectic basis to partition the Pauli group into the direct sum

$$\mathrm{P}^n = S \oplus L \oplus R, \tag{1.45}$$

so that for any $\boldsymbol{a} \in \mathrm{P}^n$, we can write $\boldsymbol{a} = \boldsymbol{a}_S + \boldsymbol{a}_L + \boldsymbol{a}_R$ for $\boldsymbol{a}_S \in S$, $\boldsymbol{a}_L \in L$, and $\boldsymbol{a}_R \in R$.

The *logical operators* of the code are elements of the quotient group $C(S)/S$. Our symplectic reduction results show that this quotient group behaves like the Pauli group on the k logical qubits, with stabilisers inherited from the logical stabiliser group L_S. Often the logical stabiliser group generators $\boldsymbol{s}_{n-k+1}, \ldots, \boldsymbol{s}_n$ are written as $\bar{Z}_1, \ldots, \bar{Z}_k$, and similarly the logical destabiliser group generators $\boldsymbol{r}_{n-k+1}, \ldots, \boldsymbol{r}_n$ are written as $\bar{X}_1, \ldots, \bar{X}_k$, because they define the logical Z and X operators on the k logical qubits. This is the sense in which a stabiliser code is said to encode k logical qubits in n physical qubits.

We can also define the *distance* of the code to be the minimum of the weights of the non-trivial logical operators

$$d = \min_{\boldsymbol{a} \in C(S) \setminus S} |\boldsymbol{a}|, \tag{1.46}$$

noting that $C(S) \setminus S$ is a set difference, rather than the quotient group $C(S)/S$, to ensure that we only obtain non-trivial logical operators.

Finally, we can consider the performance of the stabiliser code in the presence of noise. Suppose the n physical qubits are acted upon by some n-qubit Pauli channel $\mathcal{E}$ defined by its 4^n Pauli error probabilities $p_{\boldsymbol{a}}$ for all $\boldsymbol{a} \in \mathrm{P}^n$. Further suppose that some *physical error* $\boldsymbol{e} \in \mathrm{P}^n$ occurs, which we can write as $\boldsymbol{e} = \boldsymbol{e}_S + \boldsymbol{e}_L + \boldsymbol{e}_R$. Now perform a projective measurement of each of the stabiliser generators $\{\boldsymbol{s}_1, \ldots, \boldsymbol{s}_{n-k}\}$ to obtain syndromes $s_j \in \mathbb{Z}_2$ which are 0 if the measurement outcome of $\boldsymbol{s}_j$ is $+1$, and 1 if the outcome is instead -1. There is exactly one element of our symplectic basis that anticommutes with $\boldsymbol{s}_j$, namely $\boldsymbol{r}_j$. Thus the syndrome measurement yields the *error syndrome* $\boldsymbol{e}_R = s_1\boldsymbol{r}_1 + \cdots + s_{n-k}\boldsymbol{r}_{n-k}$.

The problem of *decoding* the code is, given the noise channel $\mathcal{E}$ and error syndrome $\boldsymbol{e}_R$, to determine a *recovery operator* $\boldsymbol{f} \in \mathrm{P}^n$ which is equivalent to the error $\boldsymbol{e}$ up to stabilisers, namely $\boldsymbol{f} = \boldsymbol{e} + \boldsymbol{s}'$ for some $\boldsymbol{s}' \in S$. If the decoder succeeds in determining such a recovery operator, then when the recovery operator $\boldsymbol{f}$ is applied to the physical qubits, it corrects any logical errors that may have been induced by the physical error $\boldsymbol{e}$. Otherwise, if it fails a *logical error* occurs, specified by the logical component of $\boldsymbol{e} + \boldsymbol{f}$.

First consider the conditions on the error $\boldsymbol{e}$ that guarantee successful decoding, which are known as the quantum error correction conditions. Suppose that the errors $\boldsymbol{e}$ are instead drawn from some known *error set* $E \subseteq \mathrm{P}^n$ which, importantly, is not generally a subspace. Then for any error $\boldsymbol{e} \in E$, choose an arbitrary recovery operator $\boldsymbol{f} \in E$ with the same error syndrome, $\boldsymbol{f}_R = \boldsymbol{e}_R$, so that $\boldsymbol{e} + \boldsymbol{f} = \boldsymbol{s}' + \boldsymbol{l}'$ for some $\boldsymbol{s}' \in S$ and $\boldsymbol{l}' \in L$. Decoding always succeeds if it is always the case that $\boldsymbol{l}' = \boldsymbol{0}$. Referencing the notation in Equation 1.46, we see that decoding always succeeds for an error set E if $\boldsymbol{e} + \boldsymbol{f} \notin C(S) \setminus S$ for all $\boldsymbol{e}, \boldsymbol{f} \in E$ such that $\boldsymbol{e}_R = \boldsymbol{f}_R$. This implies that the code can correct all errors of weight at most $\lfloor (d-1)/2 \rfloor$. However, the error set E can in general be exponentially large, making it intractable to naively classically search for a recovery operator $\boldsymbol{f} \in E$ satisfying the condition $\boldsymbol{f}_R = \boldsymbol{e}_R$. Classically efficient decoders are crucial for implementing quantum error correction in practice.

The optimal decoding strategy is called *maximum-likelihood* decoding, which determines the most probable logical error. Specifically, we determine the most probable logical $\boldsymbol{l}' \in L$, as assessed by the Pauli error probabilities of the Pauli channel $\mathcal{E}$, given the error syndrome $\boldsymbol{r}$. This involves computing the sums of Pauli error probabilities for stabiliser group cosets, which we write as

$$\boldsymbol{l}' = \arg\max_{\boldsymbol{m} \in L} \sum_{\boldsymbol{t} \in S} p_{\boldsymbol{t}+\boldsymbol{m}+\boldsymbol{r}}. \tag{1.47}$$

Then the recovery operator can be chosen arbitrarily from the stabiliser group coset $\boldsymbol{f} \in \boldsymbol{l}' + \boldsymbol{r} + S$. In general, maximum-likelihood decoding is expensive as it requires evaluating 4^k sums over 2^{n-k} Pauli error probabilities, though for some stabiliser codes it can be done efficiently [43].

Another decoding strategy is *minimum-weight* decoding, which determines the most probable physical error. Specifically, we determine the most probable stabiliser and logical $\boldsymbol{s}' + \boldsymbol{l}' \in S \oplus L$, as assessed by the Pauli error probabilities of the Pauli channel $\mathcal{E}$, given

the error syndrome $\boldsymbol{r}$. This involves searching for the largest Pauli error probability consistent with $\boldsymbol{r}$, which we write as

$$\boldsymbol{s}' + \boldsymbol{l}' = \arg \max_{\boldsymbol{t} \in S, \boldsymbol{m} \in L} p_{\boldsymbol{t}+\boldsymbol{m}+\boldsymbol{r}}. \tag{1.48}$$

Then the recovery operator is simply $\boldsymbol{f} = \boldsymbol{s}' + \boldsymbol{l}' + \boldsymbol{r}$. Minimum-weight perfect matching (MWPM) decoders solve this problem by mapping the decoding problem onto graph problems for X- and Z-type Pauli errors, which are then solved with Edmonds' blossom algorithm [44]. In practice, this can be very fast [45].

The performance of a decoder depends on accurate knowledge of the Pauli noise channel $\mathcal{E}$, which informs the decoder of the likelihood of different errors. This is called the *decoder prior*, and appropriately calibrating this prior is important for maximising decoder performance. The scalable Pauli noise characterisation methods we describe in Chapter 2 enable calibration of the decoder prior. Moreover, in Chapter 3 we will show that these noise estimates can be used to improve decoder performance.

1.3.3 Topological codes

Now we are ready to introduce a class of codes known as topological codes, drawing on [2]. Although there exist topological codes that are not stabiliser codes, we will only consider stabiliser topological codes here, which we will simply call topological codes.

First, we will define some classes of stabiliser codes. A quantum *low-density parity check* (LDPC) code is a family of stabiliser codes where each stabiliser generator has weight $\mathcal{O}(1)$, and each qubit is acted upon by $\mathcal{O}(1)$ stabiliser generators, each independent of the number of qubits in the code n [46]. Then a *D-dimensional local code* is a quantum LDPC code whose qubits can be arranged in a D-dimensional lattice such that each stabiliser generator is contained in a ball of radius $\mathcal{O}(1)$, again independent of the number of qubits in the code n. For our purposes, a *topological code* is exactly a D-dimensional local code.

Topological codes, particularly for $D = 2$ but also for $D = 3$, are convenient for practical implementation because they can be naturally implemented by a quantum device with only local gate connectivity. Classical computing chips are generally planar for ease of manufacturing, and planar quantum chips are naturally suited to operating topological codes in two dimensions. There is considerable interest in these topological codes, though locality substantially constrains achievable performance in comparison to quantum LDPC codes.

Surface codes are an attractive and well-studied family of topological codes [5, 49–51]. For simplicity, we will only consider surface codes in two dimensions and focus on rotated surface codes, which are more efficient than unrotated surface codes. A *surface code* encodes a single logical qubit in a $d_Z \times d_X$ rectangular array of physical qubits called *data qubits*. The stabiliser generators are alternating weight-4 products of Pauli X or Pauli Z on the plaquettes, or faces, of the array, alongside weight-2 products on the boundary, and are measured with $d_Z \times d_X$ *measure qubits* placed in the centres of these plaquettes. The logical Pauli Z (X) generator can be chosen to be any straight string of Pauli Z (X) operators of length d_Z (d_X) that terminates in a boundary Z (X) stabiliser generator, which is checked by the X (Z) stabiliser generators. Figure 1.1 shows a small rotated surface code and its stabiliser generators, as well as its syndrome extraction circuit. A *syndrome extraction circuit* measures the stabilisers of a code by entangling data qubits

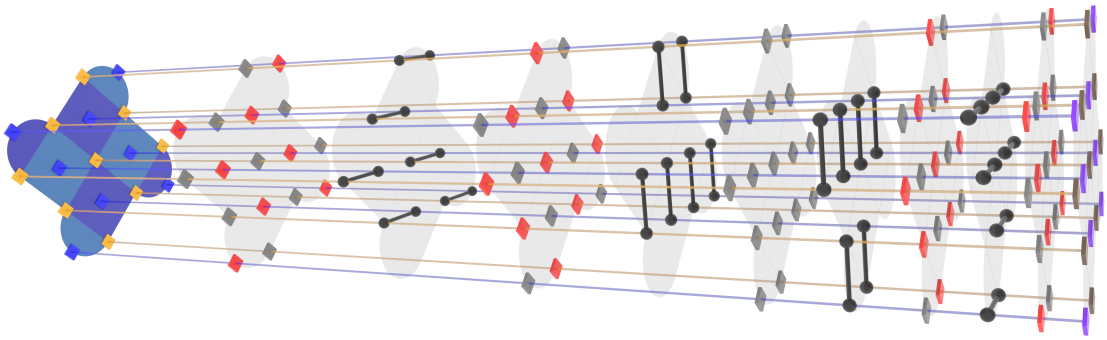

Figure 1.1: Diagram of the syndrome extraction circuit for a distance-3 surface code with 17 qubits. The circuit implements the XZZX variant of the surface code [47, 48], following recent experiments [11, 16]. The code consists of 9 data qubits (yellow) and 8 measure qubits (blue), and each measure qubit is associated with a coloured plaquette representing the Z (light blue) or X (dark blue) stabiliser it measures. The circuit is divided into 10 gate layers (light grey) implemented sequentially in time, which feature controlled-Z gates (black), Hadamard gates (red), dynamical decoupling Pauli X gates (grey), measure and reset gates (purple), and measurement dynamical decoupling gates (brown).

with adjacent measure qubits and then measuring the measure qubits. There also exist unrotated surface codes, which are similar to rotated surface codes but require more qubits to achieve the same distance.

Surface codes have a logical Z distance d_Z, and a logical X distance d_X, leading to an overall distance $d = \min(d_Z, d_X)$. While this distance is maximised for a given qubit count by a square code with $d = d_Z = d_X$, a rectangular code may perform better in practice. For example, we may be able to identify that noise in our quantum device is biased towards Pauli Z noise, either through noise characterisation or by deliberately engineering the device. Then we may be able to design syndrome extraction circuits that preserve this noise bias such that code performance is improved by increasing d_Z, which controls resistance to X noise, relative to d_X, which controls resistance to Z noise. For simplicity, however, we will now only consider square codes with distance d.

Surface codes are conducive to decoding. Maximum-likelihood decoding is possible in time quadratic in the number of qubits [43], and MWPM decoders, such as [45], are fast and perform well in practice. This leads to high thresholds and low logical error rates. Roughly, the *threshold* of a code describes the physical error rate below which the logical error rate is exponentially suppressed by increasing the code size. Consider a local stochastic noise model with physical error rate p, and denote the logical error rate $\bar{p} = \bar{p}(p, d)$. Then the *pseudo-threshold* is $p_c(d) = \bar{p}(p_c, d)$, the physical error rate that produces an equivalent logical error rate at distance d. The *asymptotic threshold*, often simply referred to as the threshold, is $\lim_{d\to\infty} p_c(d)$. Below threshold, the logical error rate is well-described by the approximate scaling relation $\bar{p} \propto \Lambda^{-d/2}$, where the *error suppression factor* Λ depends on the noise model, as well as the code, syndrome extraction circuit, and decoder. Improving all of these aspects is important when implementing quantum error correction in a device.

An aspect of surface codes that may be challenging to realise in certain hardware

architectures is the degree-4 connectivity requirement, as measure qubits need to be entangled with 4 surrounding data qubits. One proposed code that mitigates this challenge is the heavy hexagon code [52], which we can think of as a modified surface code where some measure qubits are missing, such that most qubits have degree-2 connectivity while the rest have degree-3 connectivity. This is a particular type of stabiliser code called a subsystem code [53], which measures some of the logical qubits to determine the error syndrome of the stabiliser generators with lower-weight measurements.

A *subsystem* code is a stabiliser code which splits part of the logical group into the *gauge logical group* containing the logical stabiliser and destabiliser generators for m logical qubits, $L_G = \langle \boldsymbol{s}_{n-k+1}, \ldots, \boldsymbol{s}_{n-k+m}, \boldsymbol{r}_{n-k+1}, \ldots, \boldsymbol{r}_{n-k+m} \rangle$. The logical group becomes the remaining logical generators. Then define the *gauge group* $G = S \oplus L_G$. We perform quantum error correction by measuring the generators of G, being careful with the measurement order for these non-commuting operators, in order to determine the syndromes of the stabiliser generators of S. This code encodes $k - m$ logical qubits, as those qubits are not affected by this procedure. While subsystem codes can be easier to implement by enabling reduced connectivity requirements for quantum devices, this tends to come at the cost of reduced code performance.

The heavy hexagon code does not have a threshold, that is, its asymptotic threshold is zero. However, this fact alone does not rule out the usefulness of the code, particularly in the context of near-term experiments, and we discuss a more holistic perspective on this in the next section.

1.3.4 Fault-tolerant circuits

Fault-tolerant quantum computers perform quantum computations by implementing *fault-tolerant circuits* which perform fault-tolerant logical operations protected by quantum error correcting codes. Recent experimental progress has produced devices capable of quantum error correction below the surface code threshold [16], and logical operations in the colour code [54], a class of topological codes that are more convenient for logic but have lower thresholds than surface codes. The circuits used to implement these experiments substantially influence their performance, leading to comparisons between different circuits that implement the same code [55].

I believe this represents the emergence of a new 'circuit-forward' paradigm for quantum error correction which focuses on designing fault-tolerant circuits. The syndrome extraction circuits of quantum error correcting codes form a core component of these fault-tolerant circuits. I contrast this with a 'code-forward' paradigm which focuses on designing quantum error correcting codes and treats these circuits as a relatively unimportant implementation detail. This circuit-forward paradigm has been enabled and driven by open-source software tools, particularly the stabiliser circuit simulation package Stim [56] and its interface Crumble, as well as the MWPM decoding package PyMatching [45, 57], which interfaces with Stim. These tools enable fast and easy design and simulation of fault-tolerant circuits.

These software tools frame quantum error correction in terms of detectors in quantum error correction circuits, a framing I call the *detector formalism* that is described in greater detail in [58]. A *detector* is a parity of measurement outcomes in a quantum error correction circuit that is deterministic in the absence of noise, whose deterministic outcome is set to be 0. A *logical observable* is a parity of measurement outcomes whose value corresponds to the measurement outcome of a logical Pauli operator.

Consider the example of a Z (X) memory experiment quantum error correction circuit, which is fault-tolerant. This memory experiment prepares the data qubits in the Pauli Z (X) basis, performs some number of rounds of the syndrome extraction circuit, for example as displayed in Figure 1.1, and then measures the data qubits in the Pauli Z (X) basis. In the first round of syndrome extraction, the detectors are the measurement outcomes of each Pauli Z (X) stabiliser measure qubit. Then in the following rounds, the detectors are the parities of the measurement outcomes of each measure qubit across consecutive rounds, including both Pauli Z and X stabilisers. Finally, in the last round of syndrome extraction, the detectors are the parities of the measurement outcomes of each Pauli Z (X) stabiliser measure qubit and the associated data qubits. The logical observable corresponds to the parity of data qubits along any logical Pauli Z (X) generator.

As these quantum error correction circuits are stabiliser circuits, stabiliser simulation allows a circuit-level Pauli noise model to be transformed into a *detector error model* describing the error probabilities of all possible combinations of detectors and logical observables. The fault tolerance of a quantum error correction circuit can be assessed from this detector error model [58]. It also contains the decoder prior on the likelihood of physical error configurations and enables fast simulation of the circuit to assess performance. Given the detector error model, decoding of a quantum error correction circuit proceeds from the detector and logical observable measurement results.

The detector formalism enables the co-design of quantum error correcting codes, decoders, fault-tolerant circuits, and quantum devices to jointly optimise codes, decoders, circuits, and devices given the particular strengths and weaknesses of the hardware platform upon which the devices are built. This co-design process can be informed by characterisation of circuit-level noise, which can then serve as a starting point for numerical simulations that identify the sensitivity of fault-tolerant circuits to variations in the performance of circuit components. These results can guide device engineering and inspire the design of fault-tolerant circuits for a code that mitigate the weaknesses and leverage the strengths of a device, and this is discussed further in Chapter 5. Noise characterisation results can also be used to calibrate the decoder prior, as we investigate in Chapter 3.

Chapter 2

Scalable noise characterisation of syndrome extraction circuits with averaged circuit eigenvalue sampling

> To help my all-too-human senses and brain, they translate the problem into yosegi: opening a Japanese trick box. The quantum protocols are sensations, imperfections and valleys in the marquetry, pressure points inside the wood like tense muscles, faint grins of sliding sections. I need to find the right sequence that opens it.
>
> Except that here, the trick is *not* opening it too early, the wood patterns are hidden in the countless qubits inside—each zero and one at the same time—and the moves are quantum logic operations, executed by the arrays of lasers and interferometers the gogols have built in the ship's wings. It all amounts to what the ancients called quantum process tomography: trying to figure out what the Box does to the probe states we ease into it, gently, like lockpicks. It feels like trying to juggle eight-side Rubik's cubes while trying to solve them at the same time.
>
> — Hannu Rajaniemi, *The Fractal Prince*

Abstract

Characterising the performance of noisy quantum circuits is central to the production of prototype quantum computers and can enable improved quantum error correction that exploits noise biases identified in a quantum device. We develop a scalable noise characterisation protocol suited to characterising the syndrome extraction circuits of quantum error correcting codes, a key component of fault-tolerant architectures. Our protocol builds upon averaged circuit eigenvalue sampling (ACES), a framework for noise characterisation experiments that simultaneously estimates the Pauli error probabilities of all gates in a Clifford circuit and captures averaged spatial correlations between gates implemented simultaneously in the layers of the circuit. By rigorously analysing the

performance of noise characterisation experiments in the ACES framework, we derive a figure of merit for their expected performance, allowing us to optimise their experimental design and improve the precision to which we estimate noise given fixed experimental resources. We demonstrate the scalability and performance of our protocol through circuit-level numerical simulations of the entire noise characterisation procedure for the syndrome extraction circuit of a distance-25 surface code with over 1000 qubits. Our results indicate that detailed noise characterisation methods are scalable to near-term quantum devices. We release our code in the form of the Julia package QuantumACES.

2.1 Introduction

Noise in quantum devices is the key obstacle to large-scale quantum computation. Detailed noise characterisation methods are essential for demonstrating the correct performance of prototype quantum computers, and can improve the quantum error correction required for reliable performance. We therefore seek methods capable of scaling to early fault-tolerant quantum computers [59]. Tomographic methods, such as those implemented in [60, 61], as well as gate set tomography [62], attempt to fully characterise noise processes and scale poorly as a result. This has motivated a large literature on more efficient noise characterisation methods, including compressed sensing tomography [63] and shadow tomography [64]. Randomised benchmarking and its variants [35, 65–70] are much more efficient but do not characterise noise in detail. There are a number of related methods that fit a reduced set of noise parameters within a restricted class of noise models, such as Pauli channel estimation techniques [35, 71–74], cycle benchmarking [75], and cycle error reconstruction [76].

Scalable noise characterisation methods can improve the performance of error correction by identifying which physical configurations of errors are more and less likely [77]. Noise characterisation methods can also identify biased noise [78], to which codes [48, 79, 80], decoders [81–84], and fault-tolerant operations [85], can be tailored in order to improve their performance.

Recently, rapid experimental progress has begun to demonstrate the utility of fault tolerance and quantum error correction [8–19]. Consequently, there is an increasing need for noise characterisation methods suitable for near-term quantum devices with hundreds of qubits.

In this paper, we develop a scalable noise characterisation protocol suited to characterising the syndrome extraction circuits of quantum error correcting codes by building upon averaged circuit eigenvalue sampling (ACES) [74]. ACES is a general framework for scalable noise characterisation experiments that simultaneously estimates the Pauli channels associated with the operation of all gates in a Clifford circuit, with a recent implementation effort [86]. We demonstrate our protocol on simulated data at scales of over 1000 qubits in experimentally-relevant noise parameter regimes.

Our protocol leverages the structure of fault-tolerant logical circuits. Fault-tolerant quantum computation [6, 27–29] replaces physical state preparations, gates, and measurements with redundantly encoded logical equivalents that entail regularly measuring the parity checks of a quantum error correcting code. In a typical fault-tolerant architecture, the syndrome extraction circuits that measure parity checks represent the bulk of the physical qubits and gate operations, rendering them the key target for noise characterisation experiments. We leverage the fact that the simple structures of the syndrome

extraction circuits of topological quantum codes remain similar across code sizes. As syndrome extraction circuits are representative of fault-tolerant gadgets, we believe our methods are capable of characterising other fault-tolerant gadgets such as magic state distillation [87].

In Section 2.2, we offer a high-level overview of our modified formulation of ACES, which naturally describes scalable and performant noise characterisation of the syndrome extraction circuits of topological quantum error correcting codes such as the surface code [5, 49–51]. A detailed overview follows in Section 2.3, where we rigorously analyse the performance of ACES experiments to derive a figure of merit for their experimental design as well as an improved noise estimation procedure. Then Section 2.4 describes the experimental design techniques enabling our protocol, including routines that leverage our figure of merit to optimise experimental designs for the average error rates we expect to encounter in a device.

We present numerical results in Section 2.5 for the surface code syndrome extraction circuit, which demonstrate that the performance of optimised experimental designs is asymptotically independent of the code distance and robust to differing error rates or a different noise model. We close by validating our methods in circuit-level numerical simulations of the noise characterisation of the syndrome extraction circuit of a distance-25 surface code with 1249 qubits, performed on a laptop, and release our code in the form of the Julia package QuantumACES [88]. Finally, we provide concluding remarks in Section 2.6. Our results demonstrate that ACES is capable of scalable and performant characterisation of circuit-level Pauli noise, enabling greater understanding of quantum devices, and paving the way for noise-aware decoders and tailored quantum error correcting codes.

2.2 Noise characterisation at scale

We aim to formulate a protocol, built upon ACES, that naturally describes noise characterisation experiments for the syndrome extraction circuits of topological quantum codes. We focus on surface codes, which encode a logical qubit into a $d_Z \times d_X$ rectangular array of physical qubits called *data qubits*. The stabilisers are mutually commuting Z and X parities of clusters of data qubits. They are measured by a syndrome extraction circuit, which entangles each of the $d_Z d_X - 1$ *measure qubits* with adjacent data qubits. Recent experimental progress has demonstrated improved average performance of a surface code with increasing distance [11, 16]. We will focus on the syndrome extraction circuit used in this experiment, which is depicted in Figure 2.1 for a square ($d = d_Z = d_X$) distance-3 surface code.

The syndrome extraction circuit is organised as a series of layers implemented sequentially in time. Each layer consists of a set of *gates*, relatively primitive operations on the quantum device that act on a bounded $\mathcal{O}(1)$ number of qubits, independent of the total number of qubits n in the code. The *layers* are chosen such that gates in a layer are simultaneously physically implemented by the quantum device and act on disjoint sets of qubits, ensuring that gates in a layer trivially commute with each other. What we call layers are referred to elsewhere as cycles [37, 75, 76], or gates [62]. For example, the surface code syndrome extraction circuit depicted in Figure 2.1 is divided into 9 layers consisting of controlled-Z gate layers interspersed between layers of Hadamard and dynamical decoupling X gates.

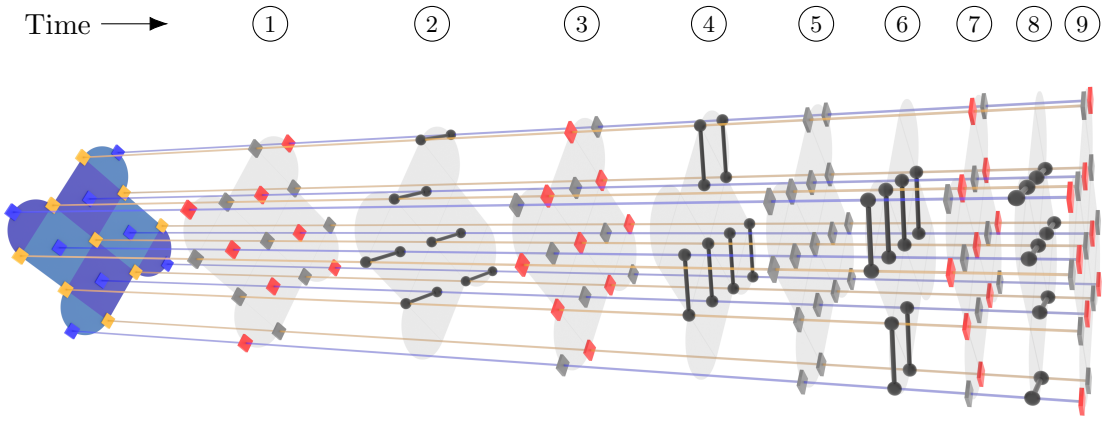

Figure 2.1: Diagram of the syndrome extraction circuit for a distance-3 surface code with 17 qubits. The circuit implements the XZZX variant of the surface code [47, 48], and its structure follows a recent experiment [11]. The code consists of 9 data qubits (yellow) and 8 measure qubits (blue), and each measure qubit is associated with a coloured plaquette representing the Z (light blue) or X (dark blue) stabiliser it measures. The circuit is divided into 9 numbered gate layers (light grey) implemented sequentially in time, which feature controlled-Z gates (black), Hadamard gates (red), and dynamical decoupling Pauli X gates (grey). Layers 1 and 9 are identical, as are layers 3 and 7. We learn a circuit-level noise model, namely single- and two-qubit Pauli channels, depending on the gate, operated in the context of its layer, as well as single-qubit Pauli channels for idle qubits in each layer.

Quantum noise can be roughly divided into coherent errors, which interfere with themselves and can accumulate, and incoherent noise, which cannot. Coherent errors can be mitigated by improved device calibration or methods such as dynamical decoupling [89]. They can also be tailored into incoherent Pauli noise by quantum error correction [38–40], as well as Pauli frame randomisation [36, 90], which can for example be realised by the randomised compiling protocol [37, 91]. Indeed, these randomisation techniques can improve the performance of quantum error correction in the presence of coherent errors [92].

We will learn a circuit-level incoherent Pauli noise model by characterising the noise associated with each gate in the circuit when implemented in the context of the layer in which it appears. This captures averaged spatial correlations between each gate and the other gates in its layer. We assume that the noise associated with each layer is independent of the prior and subsequent layers, as well as the time at which the layer was implemented. This type of noise is often referred to in benchmarking as Markovian and time-stationary [35, 69]. Dynamical decoupling pulses are believed to improve the reliability of this assumption, and we observe that the dynamical decoupling layer 5 in Figure 2.1 ensures that all two-qubit gate layers are separated by single-qubit gate layers. Also, the noise associated with the X gates in layer 5 is learned separately from the noise associated with their counterparts acting on the same qubits in layers 1, 3, 7, and 9.

At its core, ACES estimates the noise associated with a set of circuits constructed by rearranging the layers of the original circuit in order to infer the noise associated with each of the gates in the original circuit. These rearrangements are more precisely permutations with repetition, or *tuples* T, which will allow us to parameterise the design

of a noise characterisation experiment by the *tuple set* $\mathcal{T}$. Tuples can be of arbitrary length, and their elements are drawn from the *unique layer indices* $\mathcal{I}$. For example, in Figure 2.1 layers 1 and 9 and layers 3 and 7 are identical, so $\mathcal{I} = \{1, 2, 3, 4, 5, 6, 8\}$, and the tuple for the original circuit is $T = (1, 2, 3, 4, 5, 6, 3, 8, 1)$.

We focus on tuples because the structure of the surface code syndrome extraction circuit is similar regardless of the code size. This is entailed by two properties, namely that the surface code is a quantum low-density parity-check (LDPC) code, and the stabilisers are geometrically local. Specifically, the stabilisers of quantum LDPC codes include a constant $\mathcal{O}(1)$ number of qubits, and their qubits are included in a constant $\mathcal{O}(1)$ number of stabilisers [46]. Topological quantum codes have these properties, which allows us to divide their syndrome extraction circuits into a number of layers l, the circuit depth, that is independent of the number of qubits in the code n.

Consequently, we will see that the performance of a tuple set is empirically precisely predictable as a function of the surface code distance d. This will allow us to optimise tuple sets according to their performance at small code distances, and then use the optimised experimental design to characterise the noise associated with large surface codes for which direct optimisation is infeasible. Both the noise estimation precision and the number of experiments required on the quantum device are asymptotically independent of the surface code distance d. This achieves our goal of performance at scale.

ACES is a flexible framework capable of describing noise characterisation experiments that learn more detailed spatial and temporal correlations than we consider here. However, learning such correlations trades off against the performance and scalability of the protocol. While our numerical results focus on a syndrome extraction circuit for the *rotated* surface code, which we simply call the surface code, Section 2.H shows that our methods perform remarkably similarly for the *unrotated* surface code. We believe that our methods are capable of characterising the noise associated with the syndrome extraction circuits of topological quantum codes, as well as other fault-tolerant gadgets such as magic state distillation [87]. Practical architectures for quantum LDPC codes will need to address the issue of measuring non-local stabilisers, which may also allow our methods to extend to these codes.

2.3 ACES overview

We now overview ACES, recasting the original presentation of [74] to instead view circuits as sequences of layers. This allows us to parameterise noise characterisation experiments by a set of tuples that rearrange these circuit layers, a natural framing for the syndrome extraction circuits of topological quantum codes.

Then we analyse the eigenvalue estimation procedure of [93] in the context of simultaneous estimation of multiple eigenvalues. This allows us to improve upon the core least squares noise estimation procedure given in [74], and derive a figure of merit for the expected performance of ACES noise characterisation experiments which we later use to optimise their experimental design.

2.3.1 Mathematical preliminaries

We begin with a few mathematical preliminaries. The n-qubit *Pauli group* $\mathbf{P}^n$ contains all n-fold tensor products of Pauli operators $P_{\boldsymbol{a}}$ indexed by length $2n$ bit strings $\boldsymbol{a} =$

$(\boldsymbol{a}^{(x)}, \boldsymbol{a}^{(z)}) = (a_1^{(x)}, \ldots, a_n^{(x)}, a_1^{(z)}, \ldots, a_n^{(z)})$ such that

$$P_{\boldsymbol{a}} = \bigotimes_{j=1}^{n} i^{a_j^{(x)} a_j^{(z)}} X^{a_j^{(x)}} Z^{a_j^{(z)}}. \tag{2.1}$$

Indeed, the group contains all $\zeta P_{\boldsymbol{a}}$, where $\zeta \in \{\pm 1, \pm i\}$ is an overall phase factor, though we generally write $P_{\boldsymbol{a}} \in \mathbf{P}^n$ to refer to the unit phase element of the group. The group operation is matrix multiplication, and corresponds, up to a phase factor, to bit string addition

$$P_{\boldsymbol{a}} P_{\boldsymbol{a}'} = \zeta(\boldsymbol{a}, \boldsymbol{a}') P_{\boldsymbol{a}+\boldsymbol{a}'}. \tag{2.2}$$

It is therefore convenient to refer to Paulis $P_{\boldsymbol{a}}$ by their bit string $\boldsymbol{a}$, and we will often write $\boldsymbol{a} \in \mathbf{P}^n$. The group commutation relations are given by the symplectic bilinear form $\omega(\boldsymbol{a}, \boldsymbol{a}') = \boldsymbol{a}^{(x)} \cdot \boldsymbol{a}'^{(z)} + \boldsymbol{a}^{(z)} \cdot \boldsymbol{a}'^{(x)}$ as

$$P_{\boldsymbol{a}} P_{\boldsymbol{a}'} = (-1)^{\omega(\boldsymbol{a}, \boldsymbol{a}')} P_{\boldsymbol{a}'} P_{\boldsymbol{a}}, \tag{2.3}$$

which is also symmetric as we work over $\mathbb{Z}_2$. Finally, define the *support* of a Pauli $\boldsymbol{a}$ to be the set of qubits on which that Pauli is not the identity, that is,

$$\operatorname{supp}(\boldsymbol{a}) = \{j \in [n] \colon \{a_j^{(x)}, a_j^{(z)}\} \neq \{0, 0\}\}, \tag{2.4}$$

where we adopt the convention $[n] = \{1, \ldots, n\}$. A Pauli supported on b qubits is also called a weight-b Pauli.

A *Pauli channel* (for example, see [94]) is a quantum channel that can be represented in the form

$$\mathcal{E}(\rho) = \sum_{\boldsymbol{a} \in \mathbf{P}^n} p_{\boldsymbol{a}} P_{\boldsymbol{a}} \rho P_{\boldsymbol{a}}, \tag{2.5}$$

where the *Pauli error probabilities* $p_{\boldsymbol{a}}$ form a probability distribution over Pauli operators. We seek to characterise noise represented in the form of Pauli channels by estimating the Pauli error probabilities of those channels. To this end, consider the *Pauli channel eigenvectors*, which are simply the Pauli operators

$$\mathcal{E}(P_{\boldsymbol{a}'}) = \lambda_{\boldsymbol{a}'} P_{\boldsymbol{a}'}. \tag{2.6}$$

From Equations 2.3, 2.5 and 2.6, we see that the *Pauli channel eigenvalues* $\lambda_{\boldsymbol{a}'}$ are related to the Pauli error probabilities $p_{\boldsymbol{a}}$ as

$$\lambda_{\boldsymbol{a}'} = \sum_{\boldsymbol{a} \in \mathbf{P}^n} (-1)^{\omega(\boldsymbol{a}, \boldsymbol{a}')} p_{\boldsymbol{a}}. \tag{2.7}$$

In fact, this relation is a *Walsh-Hadamard transform*, ordered according to the symplectic form ω rather than the usual inner product [72]. As the transform is its own inverse, up to a constant, the error probabilities can similarly be determined from the eigenvalues as

$$p_{\boldsymbol{a}} = \frac{1}{4^n} \sum_{\boldsymbol{a}' \in \mathbf{P}^n} (-1)^{\omega(\boldsymbol{a}, \boldsymbol{a}')} \lambda_{\boldsymbol{a}'}. \tag{2.8}$$

We will focus on estimating the Pauli channel eigenvalues, as they recover the Pauli error probabilities and are easier to estimate.

Define the n-qubit *Clifford group* $\mathbf{C}^n$ to be the group generated by the controlled-X, Hadamard, and phase gates, written $C_i(X_j)$, H_j, and S_j, respectively, for control qubits i and target qubits $j \neq i$ [32]. Elements of the Clifford group $G \in \mathbf{C}^n$ act by conjugation on Pauli group elements $P_{\boldsymbol{a}}$ as $GP_{\boldsymbol{a}}G^\dagger = (\pm)_{G,\boldsymbol{a}} P_{G(\boldsymbol{a})}$, where the resulting Pauli $P_{G(\boldsymbol{a})}$ and sign $(\pm)_{G,\boldsymbol{a}}$ can be computed efficiently in the number of qubits n [31, 32]. We find it convenient to refer to $\mathcal{G}$ rather than G, where $\mathcal{G}(P_{\boldsymbol{a}}) = GP_{\boldsymbol{a}}G^\dagger$, and do so henceforth. In the context of surface code syndrome extraction circuits, we also encounter the generating gates of the Pauli group X_j, Y_j, and Z_j, as well as the controlled- Z gate $C_i(Z_j) = H_j C_i(X_j) H_j$. Analogously to the support of a Pauli, define the support of a Clifford group element $\mathrm{supp}\,(\mathcal{G})$ to be the set of qubits on which that element acts non-trivially.

Specifically, we will be interested in characterising Pauli noise channels $\mathcal{E}_{\mathcal{G}}$ associated with Clifford gates $\mathcal{G}$, whose noisy implementation we write as $\tilde{\mathcal{G}} = \mathcal{G}\mathcal{E}_{\mathcal{G}}$. The noisy implementation therefore acts as

$$\tilde{\mathcal{G}}(P_{\boldsymbol{a}}) = \lambda_{\mathcal{G},\boldsymbol{a}} \mathcal{G}(P_{\boldsymbol{a}}) = (\pm)_{\mathcal{G},\boldsymbol{a}} \lambda_{\mathcal{G},\boldsymbol{a}} P_{\mathcal{G}(\boldsymbol{a})}, \tag{2.9}$$

where $\lambda_{\mathcal{G},\boldsymbol{a}}$ is called the *generalised eigenvalue* of $\mathcal{G}$ acting on $P_{\boldsymbol{a}}$, associated with its Pauli noise channel $\mathcal{E}_{\mathcal{G}}$. For convenience, we will refer to generalised eigenvalues as eigenvalues.

2.3.2 ACES fundamentals

We can now present the core idea behind ACES, omitting for clarity the Pauli frame randomisation that renders it 'averaged'. Pauli frame randomisation tailors arbitrary noise channels on average into Pauli channels [36, 90], and this can be realised in arbitrary quantum circuits by the randomised compiling protocol [37, 91]. However, as we only consider Clifford circuits, generic Pauli frame randomisation suffices here, and consequently we assume that all noise channels are Pauli channels. Adding the random Pauli gates required for Pauli frame randomisation to our approach is straightforward, and we refer the reader to the original presentation of ACES for details [74].

In our presentation of ACES, we focus on characterising the Pauli noise associated with implementing a *circuit* $\mathcal{C}$ of Clifford gates acting on n qubits. The circuit is divided into a series of l *layers* $\mathcal{C}_i$ each composed of some number k_i of Clifford gates $\mathcal{G}_{ij}$. As we assume that the noise is time-independent, the noisy implementation of each layer can be written as $\tilde{\mathcal{C}}_i = \mathcal{C}_i \mathcal{E}_i$ for some Pauli noise channel $\mathcal{E}_i$.

As it is generally intractable to learn all 4^n eigenvalues of such an n-qubit Pauli channel, we learn a circuit-level noise model. This noise model does not explicitly represent noise that is spatially correlated between gates in a layer and only learns up to weight-two Pauli errors because we only consider single- and two-qubit gates. Specifically, we learn the Pauli channels $\mathcal{E}_{ij}$ associated with the gates $\mathcal{G}_{ij}$ operated in the context of each unique layer $\mathcal{C}_i$ for $i \in \mathcal{I}$, as well as single-qubit Pauli channels for idle qubits in each layer. We do this by estimating the *gate eigenvalues* $\lambda_{ij,\boldsymbol{a}}$, which yield the Pauli error probabilities of each channel by Equation 2.8. In Section 2.A, we show that the Pauli error probabilities estimated by ACES for the gate channel $\mathcal{E}_{ij}$ are the marginal of the Pauli error probabilities of the layer channel $\mathcal{E}_i$ onto the support of the gate $\mathcal{G}_{ij}$, demonstrating that ACES captures averaged spatial correlations between gates appearing in the same layer.

To specify the entire collection of gate eigenvalues, let us first write the set of Pauli

operators supported on some gate $\mathcal{G}$ as

$$\text{Pauli}(\mathcal{G}) = \{\boldsymbol{a} \in \mathbf{P}^n \colon \text{supp}(\boldsymbol{a}) \subseteq \text{supp}(\mathcal{G})\}, \tag{2.10}$$

a subgroup of $\mathbf{P}^n$. For a single-qubit gate, this consists of the Paulis I, X, Y, and Z, and in general for a gate supported on b qubits, there are 4^b Paulis in this set. Then the set of gate eigenvalue indices N is

$$N = \{(ij, \boldsymbol{a}) \colon i \in \mathcal{I}, j \in [k_i], \boldsymbol{a} \in \text{Pauli}(\mathcal{G}_{ij})\}, \tag{2.11}$$

where $i \in \mathcal{I}$ indexes the unique layers in the circuit and $j \in [k_i]$ indexes the gates in the layer $\mathcal{C}_i$. Then writing the multi-index $\nu = (ij, \boldsymbol{a})$, we seek to estimate λ_ν for all $\nu \in N$. For trace-preserving channels where leakage is absent, identity eigenvalues are always 1 and do not need to be estimated, so for b-qubit gates we need only estimate $4^b - 1$ gate eigenvalues.

Next, let us formally introduce the concept of a tuple. Write the circuit as

$$\mathcal{C} = \mathcal{C}_l \ldots \mathcal{C}_2 \mathcal{C}_1, \quad \mathcal{C}_i = \prod_{j \in [k_i]} \mathcal{G}_{ij}, \tag{2.12}$$

where we time-order the product for the circuit $\mathcal{C}$, with $\mathcal{C}_1$ indicating the first layer and earliest time. The gates $\mathcal{G}_{ij}$ in the layer $\mathcal{C}_i$ commute, so the order of the product does not matter. Then a *tuple* T of some arbitrary length L is a sequence of L numbers in $\mathcal{I}$ ordering the layers of the circuit $\mathcal{C}$ that acts to rearrange it as

$$\mathcal{C}_T = \prod_{u \in [L]} \mathcal{C}_{T_u}. \tag{2.13}$$

Figure 2.2 gives a concrete example of a three-layer circuit rearranged by the length 4 tuple $T = (2, 1, 3, 2)$.

Now we are ready to describe the circuit eigenvalues from which we estimate the gate eigenvalues. Following Equation 2.9, the *circuit eigenvalue* $\Lambda_{T,\boldsymbol{a}}$ of the noisy implementation $\tilde{\mathcal{C}}_T$ of the circuit $\mathcal{C}_T$ describes the overall effect of the noise on the Pauli $P_{\boldsymbol{a}}$ as

$$\tilde{\mathcal{C}}_T(P_{\boldsymbol{a}}) = \Lambda_{T,\boldsymbol{a}} \mathcal{C}_T(P_{\boldsymbol{a}}) = (\pm)_{T,\boldsymbol{a}} \Lambda_{T,\boldsymbol{a}} P_{T(\boldsymbol{a})}. \tag{2.14}$$

For convenience, we write $P_{T(\boldsymbol{a})}$ to refer to $P_{\mathcal{C}_T(\boldsymbol{a})}$, and similarly with the sign.

In fact, Equation 2.14 is identical to Equation 2.9, except that the circuit $\mathcal{C}_T$ can be decomposed into layers of gates, each with their own gate eigenvalues. Tracking the evolution of the Pauli as it is acted upon by the circuit allows us to determine how the circuit eigenvalue is composed as a product of gate eigenvalues, as shown in Figure 2.2. Using u to index the circuit layer time-step, write $P_{\boldsymbol{a}^{(u)}}$ for the state of the Pauli before the action of the uth layer, so that $\mathcal{C}_{T_u}$ maps $P_{\boldsymbol{a}^{(u)}}$ to $P_{\boldsymbol{a}^{(u+1)}}$, up to a sign factor. Then a simple calculation yields

$$\Lambda_{T,\boldsymbol{a}} = \prod_{u \in [L]} \prod_{j \in [k_{T_u}]} \lambda_{T_u j, \boldsymbol{a}^{(u)}}, \tag{2.15}$$

where the gate eigenvalues $\lambda_{T_u j, \boldsymbol{a}^{(u)}}$ implicitly depend only on the qubits of $P_{\boldsymbol{a}^{(u)}}$ which lie in the support of the gates $\mathcal{G}_{T_u j}$ of the uth layer $\mathcal{C}_{T_u}$.

The key insight of ACES is that the relation Equation 2.15 can be inverted, allowing us to estimate all of the gate eigenvalues using estimates of a sufficiently large and

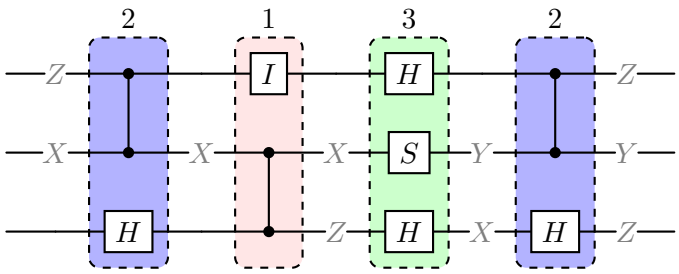

Figure 2.2: Example three-qubit circuit arranged by the tuple $T = (2, 1, 3, 2)$. Each of the Clifford gates $\mathcal{G}_{ij}$ in the circuit is associated with gate eigenvalues $\lambda_{ij,\boldsymbol{a}}$, where i labels the layer, j labels gates within a layer sequentially from the top, and $\boldsymbol{a}$ labels Paulis supported on the gate. For example, $\lambda_{32,X}$ refers to the X eigenvalue of the phase gate in layer 3, and $\lambda_{21,ZX}$ refers to the ZX eigenvalue of the controlled-Z gate in layer 2. The gates in layer 2 have the same gate eigenvalues in both occurrences. Layer 1 is padded with an identity gate. The goal of ACES is to estimate these gate eigenvalues. Labels show the propagation of the Pauli ZXI through the circuit. The corresponding circuit eigenvalue is $\Lambda_{T,ZXI} = \lambda_{21,ZX}\lambda_{12,XI}\lambda_{32,X}\lambda_{33,Z}\lambda_{21,IY}\lambda_{22,X}$. To estimate the circuit eigenvalue, prepare eigenstates of ZXI, perform the circuit, and measure in the ZYZ basis. Account for state preparation and measurement (SPAM) errors by including eigenvalues in the circuit eigenvalue product that correspond to each of the three measurements.

varied set of circuit eigenvalues. For a *tuple set* $\mathcal{T} = \{T\}$ to specify this set of circuit eigenvalues and hence the experimental design, we must supply a rule for choosing the circuit eigenvalues estimated for a tuple T. We will assume that we can prepare qubits in each of the Pauli X, Y, and Z bases, which usually requires a small number of single-qubit Clifford gates whose associated noise we do not explicitly characterise. Allowing only Z basis preparations would constrain the circuit and gate eigenvalues we are able to estimate [95]. Specifically, we will estimate circuit eigenvalues $\Lambda_{T,\boldsymbol{a}}$ corresponding to all of the Paulis $\boldsymbol{a} \in \text{Pauli}(\mathcal{G})$ supported on some gate $\mathcal{G} \in \mathcal{C}_T$ in the rearranged circuit.

This rule ensures that there exists at least one tuple set whose circuit eigenvalues are sufficient for estimating all of the gate eigenvalues. Namely, consider the tuple set $\mathcal{T}'_{\mathcal{I}} = \{(i)\}_{i\in\mathcal{I}}$ whose tuples are each of the unique layer indices $\mathcal{I}$ of the circuit repeated just once. This tuple set corresponds to direct estimation of the gate eigenvalues, as the circuit eigenvalues are precisely the gate eigenvalues and the design matrix is the identity up to the ordering of the circuit and gate eigenvalues.

Accordingly, define the *Pauli preparation set* $\mathcal{Q}_T$ as

$$\mathcal{Q}_T = \bigcup_{\mathcal{G}\in\mathcal{C}_T} \text{Pauli}(\mathcal{G}). \tag{2.16}$$

Then the set of circuit eigenvalue indices M is

$$M = \{(T, \boldsymbol{a}) \colon T \in \mathcal{T}, \boldsymbol{a} \in \mathcal{Q}_T\}. \tag{2.17}$$

Hence, writing the multi-index $\mu = (T, \boldsymbol{a})$, we will estimate the circuit eigenvalues Λ_μ for all $\mu \in M$.

Now rewrite the product over circuit layer time-steps Equation 2.15 as a product over the gate eigenvalue indices

$$\Lambda_\mu = \prod_{\nu\in N} \lambda_\nu^{A_{\mu\nu}}, \tag{2.18}$$

where $A_{\mu\nu}$ is the non-negative integer power of λ_ν as it appears in the expression for Λ_μ. As we are interested in circuit and gate eigenvalues that are not too small, lying in the interval $(\delta, 1]$ for some arbitrary $\delta > 0$, we can transform the product of Equation 2.18 into a sum by taking the (negative) logarithm. This yields circuit and gate (negative) log-eigenvalues $b_\mu = -\log \Lambda_\mu$ and $x_\nu = -\log \lambda_\nu$ which correspondingly lie in $[0, \log(1/\delta))$. Constructing the *design matrix* A elementwise from the $A_{\mu\nu}$ for each $\mu \in M$ and $\nu \in N$ obtains

$$b_\mu = \sum_{\nu \in N} A_{\mu\nu} x_\nu. \tag{2.19}$$

This is a standard linear regression problem, so we can estimate the gate log-eigenvalues from the circuit log-eigenvalues with a linear least squares method as long as the design matrix A, which is usually sparse, has full rank N. Note that we have overloaded N to refer both to the gate eigenvalue index set as well as its size, and do the same for the circuit eigenvalue index set M.

We also include gate eigenvalue parameters to account for state preparation and measurement (SPAM) errors. Owing to a gauge freedom, preparation and measurement errors cannot be estimated separately [62], a phenomenon that has also been analysed in the context of Pauli noise learning [96]. We will not treat this issue in detail, instead making the simple assumption that SPAM noise is fully associated with measurements, effectively fixing a gauge. Therefore, we model Pauli frame randomised noisy measurements with a measurement error probability p_ν corresponding to an eigenvalue $\lambda_\nu = 1 - 2p_\nu$. Here, we learn measurements in the Pauli X, Y, and Z bases separately, yielding $3n$ SPAM parameters. We could also assume that the measurement errors are the same across Pauli bases, yielding n SPAM parameters, but choose not to do so here.

With the addition of SPAM errors, the aforementioned tuple set $\mathcal{T}'_\mathcal{I} = \{(i)\}_{i \in \mathcal{I}}$ no longer produces a full-rank design matrix. We therefore augment it with the empty tuple $()$ to measure SPAM noise, obtaining the *basic tuple set* $\mathcal{T}_\mathcal{I} = \{(i)\}_{i \in \mathcal{I}} \cup \{()\}$. This tuple set constructs a full-rank design matrix for any circuit $\mathcal{C}$. We describe an algorithm for generating better-performing tuple sets in Section 2.4.3.

As we discussed in Section 2.2, the number of layers l in the syndrome extraction circuits of topological quantum codes is independent of the number of qubits in the code n. Since gates act on a bounded number of qubits, each gate has a bounded number of gate eigenvalues also independent of n, so we need only estimate $\mathcal{O}(n)$ gate eigenvalues. The basic tuple set illustrates that this only requires us to estimate $\mathcal{O}(n)$ circuit eigenvalues.

2.3.3 Circuit eigenvalue sampling

In this section, we analyse the performance of the eigenvalue sampling procedure of [93], with a focus on simultaneous estimation of circuit eigenvalues. We aim to estimate the noise associated with the syndrome extraction circuit $\mathcal{C}$ in a number of experiments that is asymptotically independent of the number of qubits n on which the circuit acts. As we need to estimate $\mathcal{O}(n)$ circuit eigenvalues, each individual experiment must also estimate $\mathcal{O}(n)$ circuit eigenvalues. As Pauli preparations and measurements supported on disjoint sets of qubits can trivially be performed simultaneously, this procedure will allow us to estimate the requisite circuit eigenvalues in $\mathcal{O}(1)$ experiments.

First, we analyse the estimation of a single circuit eigenvalue. Examining Equation 2.14 and noting that the Pauli operators form an orthogonal basis under the trace inner product,

we obtain

$$\Lambda_{T,\boldsymbol{a}} = (\pm)_{T,\boldsymbol{a}} \frac{1}{2^n} \operatorname{tr}\left(P_{T(\boldsymbol{a})} \tilde{\mathcal{C}}_T(P_{\boldsymbol{a}})\right). \tag{2.20}$$

This immediately suggests a sampling procedure for estimating the circuit eigenvalue $\Lambda_{T,\boldsymbol{a}}$, where we prepare eigenstates of $P_{\boldsymbol{a}}$, apply the noisy circuit $\tilde{\mathcal{C}}_T$, and then measure in the Pauli basis defined by $P_{T(\boldsymbol{a})}$, correcting the outcome according to the sign $(\pm)_{T,\boldsymbol{a}}$.

Specifically, consider the eigenbasis $|\psi_{\boldsymbol{s}}^{\boldsymbol{a}}\rangle$ of $P_{\boldsymbol{a}}$ consisting of all 2^n sign configurations of single-qubit Pauli eigenstate tensor products, indexed by the length n bit string $\boldsymbol{s}$. The state $|\psi_{\boldsymbol{s}}^{\boldsymbol{a}}\rangle$ is an eigenstate of $P_{\boldsymbol{a}}$ with eigenvalue s, the parity of the vector $\boldsymbol{s}$.

Suppose that we prepare such an eigenstate, with a sign configuration $\boldsymbol{s}$ chosen uniformly at random, implement the noisy circuit $\tilde{\mathcal{C}}_T$, and measure in the basis defined by $P_{T(\boldsymbol{a})}$, correcting the outcome for the signs $(\pm)_{T,\boldsymbol{a}}$ and s. The measurement outcome $m_{T,\boldsymbol{a}}$ is a Bernoulli random variable that takes values ± 1, and Equation 2.20 shows that it has expectation $\Lambda_{T,\boldsymbol{a}}$ and hence variance $1 - \Lambda_{T,\boldsymbol{a}}^2$. The sample average estimator $\hat{\Lambda}_{T,\boldsymbol{a}}$ is therefore an unbiased estimator of the circuit eigenvalue $\Lambda_{T,\boldsymbol{a}}$.

We aim to simultaneously estimate multiple circuit eigenvalues in a single experiment. In general, we could prepare eigenstates of two commuting Paulis $\boldsymbol{a}$ and $\boldsymbol{a}'$, and then measure in the commuting bases generated by $T(\boldsymbol{a})$ and $T(\boldsymbol{a}')$. However, these preparations and measurements would generally require noisy entangling gates. As we cannot fully characterise SPAM noise, we restrict ourselves to individually preparing and measuring each qubit in the Pauli X, Y, or Z basis, which requires only single-qubit Clifford gates.

This imposes a stricter requirement on simultaneously estimable circuit eigenvalues. We say that two Paulis $\boldsymbol{a}$ and $\boldsymbol{a}'$ are *T-consistent*, written $\boldsymbol{a} \diamond_T \boldsymbol{a}'$, if on each qubit of the preparations as well as the measurements, either both Paulis are the same or at least one is the identity, such that we can simultaneously estimate the circuit eigenvalues $\Lambda_{T,\boldsymbol{a}}$ and $\Lambda_{T,\boldsymbol{a}'}$. More formally, $\boldsymbol{a} \diamond_T \boldsymbol{a}'$ if $\boldsymbol{a}_j = \boldsymbol{a}'_j$ for all $j \in \operatorname{supp}(\boldsymbol{a}) \cap \operatorname{supp}(\boldsymbol{a}')$, and $T(\boldsymbol{a})_j = T(\boldsymbol{a}')_j$ for all $j \in \operatorname{supp}(T(\boldsymbol{a})) \cap \operatorname{supp}(T(\boldsymbol{a}'))$.

Then an *experiment* is a set of mutually T-consistent Pauli preparations, a subset $E_T \subset \mathcal{Q}_T$ such that $\boldsymbol{a} \diamond_T \boldsymbol{a}'$ for all $\boldsymbol{a}, \boldsymbol{a}' \in E_T$. We describe an algorithm for generating experiments in Section 2.4.1. Since all of the preparations are T-consistent, there is a unique non-identity Pauli on each qubit of the preparations and the measurements, allowing us to easily derive the corresponding n-qubit preparation $P_{\boldsymbol{A}}$ and measurement $P_{\boldsymbol{A}'}$. We write the measurement as $P_{\boldsymbol{A}'}$ because in general $P_{\boldsymbol{A}'} \neq P_{T(\boldsymbol{A})}$.

As before, randomly prepare some eigenstate $|\psi_{\boldsymbol{s}}^{\boldsymbol{A}}\rangle$ of $P_{\boldsymbol{A}}$, implement the circuit $\tilde{\mathcal{C}}_T$, and then measure each qubit in the basis defined by $P_{\boldsymbol{A}'}$. This yields a length n bit string outcome $\boldsymbol{m}_{T,\boldsymbol{A}}$, which we correct overall by the sign $(\pm)_{T,\boldsymbol{A}}$ and elementwise by $\boldsymbol{s}$. Then the parities $m_{T,\boldsymbol{a}}$ of the marginals of $\boldsymbol{m}_{T,\boldsymbol{A}}$ to $\operatorname{supp}(T(\boldsymbol{a}))$ contribute to the sample average estimators $\hat{\Lambda}_{T,\boldsymbol{a}}$ for all $\boldsymbol{a} \in E_T$.

Simultaneous measurement predictably correlates the circuit eigenvalue estimators. Consider the product of measurement outcomes $m_{T,\boldsymbol{a}} m_{T,\boldsymbol{a}'}$ for $\boldsymbol{a}, \boldsymbol{a}' \in E_T$. As $P_{\boldsymbol{a}} P_{\boldsymbol{a}'} = P_{\boldsymbol{a}+\boldsymbol{a}'}$, this is the measurement outcome of $\boldsymbol{a} + \boldsymbol{a}'$, with circuit eigenvalue $\Lambda_{T,\boldsymbol{a}+\boldsymbol{a}'}$. Then the covariance is

$$\operatorname{Cov}\left[m_{T,\boldsymbol{a}}, m_{T,\boldsymbol{a}'}\right] = \Lambda_{T,\boldsymbol{a}+\boldsymbol{a}'} - \Lambda_{T,\boldsymbol{a}} \Lambda_{T,\boldsymbol{a}'}, \tag{2.21}$$

which reduces to the variance $1 - \Lambda_{T,\boldsymbol{a}}^2$ when $\boldsymbol{a} = \boldsymbol{a}'$. Measurement outcomes can only covary within each experiment of the *experiment set* $\mathcal{E}_T = \{E_T\}$ for a tuple T, which estimates all circuit eigenvalues $\Lambda_{T,\boldsymbol{a}}$ for $\boldsymbol{a} \in \mathcal{Q}_T$.

When characterising noise on a quantum device, we aim to maximise the precision of our estimation given a fixed amount of time on the device. The time allotted to noise characterisation will be informed by factors including the timescale of device drift, the cost of device time, and the desired duty cycle balance between device characterisation and operation. Longer tuples and deeper circuits take longer to perform, so the length of the tuples in the tuple set trades off against the number of measurement shots we are able to collect.

Therefore, let us introduce some formalism to define the *measurement budget* S, a modified version of the *measurement shots* S' that accounts for the time taken to implement the circuits on the device. Suppose that the circuit corresponding to each tuple T takes some time τ_T to implement on the device. We allocate a fraction Γ_T of the shots S' to each experiment set $\mathcal{E}_T$, where the non-negative *shot weights* $\Gamma = \{\Gamma_T\}_{T\in\mathcal{T}}$ sum to unity, and divide shots evenly between the individual experiments in each experiment set. By default, we choose the shot weights $\Gamma_T \propto \tau_T^{-1}$ such that the experiments for each tuple are measured for an equal amount of time on the device. Now define the *time factor* $\tau_{\mathcal{T},\Gamma} = \sum_{T\in\mathcal{T}} \Gamma_T \tau_T$, where for the default shot weights we omit the Γ dependence to write $\tau_{\mathcal{T}}$. Then the measurement budget S is given by

$$S = S' \left(\frac{\tau_{\mathcal{T}_\mathcal{I}}}{\tau_{\mathcal{T},\Gamma}} \right), \tag{2.22}$$

and represents the number of measurement shots collected by the ACES *experimental design* $(\mathcal{T}, \Gamma)$ in the time the basic tuple set collects S' measurement shots. Lastly, let the *experiment measurement budget* $S_T = S\Gamma_T/|\mathcal{E}_T|$ be the measurement budget for each experiment in the experiment set $\mathcal{E}_T$ for the tuple T.

Finally, we are ready to specify the elements of the *circuit eigenvalue estimator covariance matrix* Ω of the circuit eigenvalue estimator vector $\hat{\mathbf{\Lambda}}$ by modifying Equation 2.21 to account for the number of experiments measuring the relevant circuit eigenvalues and the experiment duration. Let $E_{T,\boldsymbol{a}}$ be the number of experiments $E_T \in \mathcal{E}_T$ measuring the circuit eigenvalue $\Lambda_{T,\boldsymbol{a}}$, and similarly let $E_{T,\boldsymbol{a},\boldsymbol{a}'}$ be the number of experiments that simultaneously measure $\Lambda_{T,\boldsymbol{a}}$ and $\Lambda_{T,\boldsymbol{a}'}$, noting that $E_{T,\boldsymbol{a},\boldsymbol{a}} = E_{T,\boldsymbol{a}}$. Then each eigenvalue is measured with $S_T E_{T,\boldsymbol{a}}$ shots, and $\Lambda_{T,\boldsymbol{a}}$ and $\Lambda_{T,\boldsymbol{a}'}$ are simultaneously measured with $S_T E_{T,\boldsymbol{a},\boldsymbol{a}'}$ shots. This yields the expression for the covariance matrix entries

$$\mathrm{Cov}\left[\hat{\Lambda}_{T,\boldsymbol{a}}, \hat{\Lambda}_{T,\boldsymbol{a}'}\right] = \frac{E_{T,\boldsymbol{a},\boldsymbol{a}'}}{S_T E_{T,\boldsymbol{a}} E_{T,\boldsymbol{a}'}} \left(\Lambda_{T,\boldsymbol{a}+\boldsymbol{a}'} - \Lambda_{T,\boldsymbol{a}}\Lambda_{T,\boldsymbol{a}'}\right). \tag{2.23}$$

Note that the covariance matrix is block diagonal, as the circuit eigenvalue estimators for different tuples are uncorrelated, and sparse, as entries are zero when the supports of $\boldsymbol{a}$ and $\boldsymbol{a}'$ remain disjoint throughout their evolution under $\mathcal{C}_T$.

2.3.4 Estimating gate Pauli errors

We now complete the ACES noise estimation procedure by describing a method for estimating the Pauli error probabilities of each of the gates from estimates of the circuit eigenvalues. This primarily entails solving the linear regression problem posed in Equation 2.19 to estimate the gate log-eigenvalues from the circuit log-eigenvalues. We use Equation 2.23 for the circuit eigenvalue estimator covariance matrix Ω to replace the ordinary least squares method described in [74] with weighted least squares. Weighted least squares offers greater performance than ordinary least squares while remaining

practically scalable past a thousand qubits. We further discuss and compare least squares methods in Section 2.B.

Begin by casting the linear regression problem of Equation 2.19 into the standard form

$$\boldsymbol{b} = A\boldsymbol{x} + \boldsymbol{\varepsilon}, \tag{2.24}$$

where the circuit log-eigenvalues $\boldsymbol{b}$ are related to the gate log-eigenvalues $\boldsymbol{x}$ by the design matrix A. We assume the error variable is distributed according to a multivariate normal distribution $\boldsymbol{\varepsilon} \sim \mathcal{N}_M(0, \Omega')$, where Ω' is the circuit log-eigenvalue estimator covariance matrix. As the circuit log-eigenvalues are the logarithm of the circuit eigenvalues, we estimate Ω' with the circuit eigenvalue estimator covariance matrix Ω. The appropriate first-order Taylor expansion is given elementwise as $\Omega'_{\alpha\beta} \approx \Omega_{\alpha\beta}/\Lambda_\alpha\Lambda_\beta$. This approximation works well in practice and accurately predicts the performance of ACES.

The weighted least squares estimator for the gate log-eigenvalues is

$$\hat{\boldsymbol{x}} = \left(A^\intercal \hat{W} A\right)^{-1} A^\intercal \hat{W} \boldsymbol{b}. \tag{2.25}$$

The weight matrix W is a diagonal matrix consisting of the inverse circuit log-eigenvalue estimator variances $W_{\mu\mu} = \Omega'^{-1}_{\mu\mu}$. These variances are estimated with the circuit eigenvalue estimators, so we properly write this as the *estimated weight matrix* $\hat{W}$. The gate eigenvalues are at most 1, so the (negative) gate log-eigenvalues are non-negative, and we therefore set negative elements of $\hat{\boldsymbol{x}}$ to 0. In practice, we reduce weighted least squares to ordinary least squares by left-multiplying Equation 2.24 by $\sqrt{\hat{W}}$, allowing us to leverage optimised and numerically stable linear algebra routines for sparse ordinary least squares.

Finally, we estimate the Pauli error probabilities of each gate by taking the Walsh-Hadamard transform of the corresponding gate eigenvalues, following Equation 2.8. The resulting estimates are not guaranteed to be valid probability distributions, so we project them using the Euclidean distance into the probability simplex with a simple algorithm [97], completing our noise estimation procedure.

2.3.5 ACES performance analysis

We aim to maximise the precision to which we estimate noise given a fixed measurement budget S. To do this, we analyse the performance of an ACES experimental design $(\mathcal{T}, \Gamma)$, which consists of the tuple set $\mathcal{T}$ and shot weights Γ, when characterising a circuit $\mathcal{C}$ with gate eigenvalues $\boldsymbol{\lambda}$. The original presentation of ACES suggested the pseudoinverse norm of the design matrix A [74], but we use the circuit eigenvalue estimator covariance matrix described in Equation 2.23 to precisely calculate the distribution of the gate eigenvalue estimator vector $\hat{\boldsymbol{\lambda}}$ and thereby derive an improved figure of merit.

We calculate the gate log-eigenvalue estimator covariance matrix Σ' by conjugating the circuit log-eigenvalue covariance matrix Ω' by the weighted least squares estimation matrix of Equation 2.25, so that

$$\Sigma' = \left(A^\intercal W A\right)^{-1} A^\intercal W \Omega' W A \left(A^\intercal W A\right)^{-1}. \tag{2.26}$$

We calculate the figure of merit with respect to known values, and so use W rather than $\hat{W}$. As the gate eigenvalues are the exponential of the gate log-eigenvalues, the *gate eigenvalue estimator covariance matrix* Σ follows from Σ' by another first-order Taylor expansion given elementwise by $\Sigma_{\alpha\beta} \approx \lambda_\alpha \lambda_\beta \Sigma'_{\alpha\beta}$. Under our assumptions, the

gate eigenvalue estimator residual vector $\hat{\boldsymbol{\lambda}} - \boldsymbol{\lambda}$ is distributed according to a multivariate normal distribution $\mathcal{N}_N(0, \Sigma)$, as linear least squares is an unbiased estimator.

Now consider the quadratic form $(\hat{\boldsymbol{\lambda}} - \boldsymbol{\lambda})^\intercal(\hat{\boldsymbol{\lambda}} - \boldsymbol{\lambda}) = \|\hat{\boldsymbol{\lambda}} - \boldsymbol{\lambda}\|_2^2$, the squared two-norm of the gate eigenvalue residual vector. It is distributed as a weighted sum of chi-squared random variables with 1 degree of freedom χ_1^2—the distribution of the square of a standard normal random variable—where the weights σ_ν are the eigenvalues of Σ, that is

$$\|\hat{\boldsymbol{\lambda}} - \boldsymbol{\lambda}\|_2^2 = \sum_{\nu=1}^{N} \sigma_\nu y_\nu, \quad y_\nu \sim \chi_1^2. \tag{2.27}$$

This is a generalised chi-squared distribution with mean $\operatorname{tr}(\Sigma)$ and variance $2\operatorname{tr}(\Sigma^2)$. While it has no known closed-form probability density function [98], this distribution can be calculated numerically [99].

In particular, we will focus on the normalised root-mean-square (RMS) error $\sqrt{S'/N}\|\hat{\boldsymbol{\lambda}} - \boldsymbol{\lambda}\|_2$ proportional to the square root of the quadratic form. This quantity averages the error over the number of gate eigenvalues N, and is normalised to remove the dependence on the measurement shots S' implicitly introduced in Equation 2.23.

We seek a figure of merit that is easily computed and accurately predicts the performance of ACES noise characterisation experiments, which will allow us to optimise their experimental design. To this end, define the *ACES figure of merit* to be the expected normalised RMS error $\mathcal{F} = \sqrt{S'/N}\,\mathbb{E}\left[\|\hat{\boldsymbol{\lambda}} - \boldsymbol{\lambda}\|_2\right]$. Smaller figures of merit are better, as the measurement budget required to achieve a fixed estimation accuracy is proportional to the square of the figure of merit. Note this implies that ACES recovers the usual scaling for additive precision Pauli channel estimation [94].

The ACES figure of merit can be approximated by a second-order Taylor expansion as

$$\mathcal{F}(\mathcal{T}, \Gamma|\mathcal{C}, \boldsymbol{\lambda}) \approx \sqrt{S'/N}\sqrt{\operatorname{tr}(\Sigma)}\left(1 - \frac{1}{4}\frac{\operatorname{tr}(\Sigma^2)}{\operatorname{tr}(\Sigma)^2}\right). \tag{2.28}$$

Since we already know the expectation of the square from the quadratic form, the normalised RMS error variance $\mathcal{V} = S'/N\,\mathbb{V}\left[\|\hat{\boldsymbol{\lambda}} - \boldsymbol{\lambda}\|_2\right]$ follows as

$$\mathcal{V}(\mathcal{T}, \Gamma|\mathcal{C}, \boldsymbol{\lambda}) \approx \frac{S'}{2N}\frac{\operatorname{tr}(\Sigma^2)}{\operatorname{tr}(\Sigma)}\left(1 - \frac{1}{8}\frac{\operatorname{tr}(\Sigma^2)}{\operatorname{tr}(\Sigma)^2}\right). \tag{2.29}$$

These approximations are sufficiently accurate in practice that we refer to the approximations and true values interchangeably. Note that when the shot weights Γ are omitted, we use the default shot weights for $\mathcal{T}$.

Crucial to a full understanding of ACES is a focus on optimising experimental designs. Absent this, it is easy to conclude that ACES is only capable of estimating noise to additive precision [76]. Section 2.C analyses a toy problem to demonstrate that ACES is in general capable of relative precision noise estimation. However, only certain combinations of gate eigenvalues can be learned to relative precision. To use the language of cycle error reconstruction [76], the *orbits* of a gate are the sets of Paulis mapped to each other by the action of the gate, and only the products of gate eigenvalues within each of the orbits of the gate can be learned to relative precision. For example, consider the orbits of the Hadamard gate H, which are (X, Z) and (Y), and notice that repeating the gate only allows us to amplify the quantities $\lambda_{H,X}\lambda_{H,Z}$ and $\lambda_{H,Y}$ and hence estimate them to relative precision. Empirically, when examining the gate eigenvalue estimator covariance

matrix Σ for optimised experimental designs, we find that gate eigenvalue estimators within gate orbits are strongly anticorrelated, such that marginalising over gate orbits obtains a relative precision estimator, though we leave further analysis to future work.

We therefore believe that ACES performs comparably to relative precision Pauli channel estimation techniques in the literature such as cycle error reconstruction (CER) [76]. In principle, each framework could be modified to obtain the strengths of the other. For example, our presentation of ACES here focuses on learning a circuit-level noise model, but could be modified to characterise noise that is spatially correlated between gates in a layer, as in CER. However, this would come at the cost of increasing the number of experiments required to implement the protocol on a quantum device, and our choices in this paper reflect our focus on developing a practical and scalable noise characterisation protocol.

2.4 Designing ACES experiments

Our overview of ACES in Section 2.3 elided the matter of designing ACES noise characterisation experiments, which we now address. First, we describe a deterministic algorithm that generates the experimental set $\mathcal{E}_T$ for a tuple $T \in \mathcal{T}$. Then, we outline a gradient descent method for optimising the shot weights Γ associated with a tuple set $\mathcal{T}$. Lastly, we outline a coordinate descent method for optimising the lengths of tuples, and describe a greedy algorithm that adds random tuples and prunes them from the set, together optimising the tuple set $\mathcal{T}$.

Our optimisation methods minimise the figure of merit, which is closely related to the trace of the gate eigenvalue estimator covariance matrix. Consequently, they roughly aim to produce what is known in the theory of optimal experimental designs as an A-optimal design [100]. We suspect that this literature offers insights for improving the heuristic algorithms and methods described here.

2.4.1 Packing experiment sets

We now introduce a deterministic algorithm for generating the experiment set $\mathcal{E}_T$ for a tuple T. With this algorithm, the tuple set $\mathcal{T}$ and shot weights Γ fully specify the design of ACES noise characterisation experiments.

Recall from Section 2.3.3 that the experiment set $\mathcal{E}_T$ must estimate the circuit eigenvalues $\Lambda_{T,\boldsymbol{a}}$ for all Paulis in the Pauli preparation set $\boldsymbol{a} \in \mathcal{Q}_T$, namely, all Paulis supported on some gate in the rearranged circuit $\mathcal{C}_T$. Moreover, if two Paulis $\boldsymbol{a}, \boldsymbol{a}' \in \mathcal{Q}_T$ are T-consistent, written $\boldsymbol{a} \diamond_T \boldsymbol{a}'$, then we can simultaneously estimate $\Lambda_{T,\boldsymbol{a}}$ and $\Lambda_{T,\boldsymbol{a}'}$ in the same experiment.

Algorithm 1 packs the Pauli preparations $\mathcal{Q}_T$ into experiments, mutually T-consistent sets E_T, such that the experiment set $\mathcal{E}_T = \{E_T\}$ estimates all of the circuit eigenvalues associated with the tuple T. The *unadded Pauli set* $\mathcal{U}_T$ is initialised as $\mathcal{Q}_T$, and tracks the Paulis not yet added to any experiment. The algorithm adds experiments to the experiment set until it is empty. The *T-consistent unadded Pauli set* U_T and the *T-consistent Pauli preparation set* Q_T, initialised as $\mathcal{U}_T$ and $\mathcal{Q}_T$, respectively, track Paulis that can be added to a specific experiment. Each time a Pauli $\boldsymbol{a}$ is added to an experiment E_T, we update U_T and Q_T by intersecting them with the *T-consistency set* $\kappa_T(\boldsymbol{a}) = \{\boldsymbol{a}' \in \mathcal{Q}_T \colon \boldsymbol{a} \diamond_T \boldsymbol{a}'\} \setminus \{\boldsymbol{a}\}$.

Then experiments E_T are constructed by adding Paulis from U_T until it is empty, and then adding Paulis from Q_T until it is also empty. We sort the Pauli preparations in

descending order by the size of their measurement support, and add the first Pauli whose measurement has the maximum overlap with the measurements of the Paulis already in the experiment, as in general measurements can have larger supports than preparations.

Algorithm 1: Experiment set packing algorithm

Input: Pauli preparation set $\mathcal{Q}_T$
Output: Experiment set $\mathcal{E}_T$

initialise $\mathcal{E}_T \leftarrow \emptyset$
sort $\boldsymbol{a} \in \mathcal{Q}_T$ by $\left|\mathrm{supp}\left(T(\boldsymbol{a})\right)\right|$ in descending order
initialise $\mathcal{U}_T \leftarrow \mathcal{Q}_T$
compute $\kappa_T(\boldsymbol{a})$ for all $\boldsymbol{a} \in \mathcal{Q}_T$
/* Construct experiments for all Paulis */
while $\mathcal{U}_T \neq \emptyset$ **do**
    initialise $E_T \leftarrow \emptyset$
    initialise $U_T \leftarrow \mathcal{U}_T$
    initialise $Q_T \leftarrow \mathcal{Q}_T$
    /* First add Paulis not yet in any experiment */
    while $U_T \neq \emptyset$ **do**
        set $\boldsymbol{a}' \leftarrow \arg\max_{\boldsymbol{a} \in U_T} \left|\mathrm{supp}\left(T(\boldsymbol{a})\right) \cap \mathrm{supp}\left(E_T\right)\right|$
        add $E_T \leftarrow E_T \cup \boldsymbol{a}'$
        update $Q_T \leftarrow Q_T \cap \kappa_T(\boldsymbol{a}')$
        update $U_T \leftarrow U_t \cap \kappa_T(\boldsymbol{a}')$
        remove $\mathcal{U}_T \leftarrow \mathcal{U}_T \setminus \boldsymbol{a}'$
    /* Then fill the experiment with other Paulis */
    while $Q_T \neq \emptyset$ **do**
        set $\boldsymbol{a}' \leftarrow \arg\max_{\boldsymbol{a} \in Q_T} \left|\mathrm{supp}\left(T(\boldsymbol{a})\right) \cap \mathrm{supp}\left(E_T\right)\right|$
        add $E_T \leftarrow E_T \cup \boldsymbol{a}'$
        update $Q_T \leftarrow Q_T \cap \kappa_T(\boldsymbol{a}')$
    add $\mathcal{E}_T \leftarrow \mathcal{E}_T \cup \{E_T\}$
return $\mathcal{E}_T$

The eigenvalue sampling procedure requires us to prepare all eigenstate sign configurations of each Pauli in the experiment at equal frequency, which is achieved by Pauli frame randomisation. When practically implementing this protocol, one should choose the number of randomisations for the experiments $E_T \in \mathcal{E}_T$ of each tuple T in alignment with its shot weights Γ_T, perhaps imposing a minimum. Then, allocate a constant number of shots to each randomisation, optimally a single shot by leveraging techniques such as in [101].

2.4.2 Optimising measurement allocation

We aim to optimise the shot weights Γ, initialised as their default values, of a tuple set $\mathcal{T}$ with respect to its figure of merit $\mathcal{F}(\mathcal{T}, \Gamma|\mathcal{C}, \boldsymbol{\lambda})$ when characterising a circuit $\mathcal{C}$ with gate eigenvalues $\boldsymbol{\lambda}$. Specifically, we parameterise the shot weights with the *shot log-weights* $\gamma = \{\gamma_T\}_{T \in \mathcal{T}}$ as $\Gamma_T = \exp\left(-\gamma_T\right) / \sum_{U \in \mathcal{T}} \exp\left(-\gamma_U\right)$, and perform gradient descent on the shot log-weights γ.

Nesterov momentum augments gradient descent with a velocity that tracks gradients across update steps [102, 103]. The learning rate $\eta > 0$ controls the size of the contribution of the gradient to the velocity at each step, whereas the momentum coefficient $\mu \in [0, 1]$ controls the decay in the velocity across update steps. Gradient descent corresponds to $\mu = 0$. Writing the shot log-weights as a vector $\boldsymbol{\gamma}$, the update step is given by

$$\boldsymbol{v}^{(s+1)} = \mu \boldsymbol{v}^{(s)} - \eta \frac{\partial \mathcal{F}}{\partial \boldsymbol{\gamma}} \left(\mathcal{T}, \boldsymbol{\gamma}^{(s)} + \mu \boldsymbol{v}^{(s)} \right), \tag{2.30}$$

$$\boldsymbol{\gamma}^{(s+1)} = \boldsymbol{\gamma}^{(s)} + \boldsymbol{v}^{(s+1)}. \tag{2.31}$$

In practice, we find it helpful to revert update steps that worsen the figure of merit and then zero the velocity. If this happens multiple times in quick succession, we also reduce the learning rate η by a factor η_r. We usually choose parameter values $\eta = 10^{3/4}$, $\mu = 0.99$, and $\eta_r = 10^{1/4}$.

Gradient calculations are optimised with the analytic expressions in Section 2.D.

2.4.3 Optimising tuple sets

When introducing tuple sets in Section 2.3.2, we only described how to construct the basic tuple set $\mathcal{T}_{\mathcal{I}}$, which describes the most basic ACES noise characterisation experiment. We now outline two algorithms that allow us to optimise tuple sets $\mathcal{T}$ by their figure of merit.

Elfving's theorem characterises optimal designs for linear regression problems [104, 105], which use a minimal number of maximally spaced points. In the context of ACES, shallow tuples produce large circuit eigenvalues near 1, whereas deep tuples with a large number of layers produce smaller eigenvalues. We therefore augment the basic tuple set, containing shallow tuples, with the *repeated tuple set* $\mathcal{T}_{\text{rep}}$, whose construction we describe in Section 2.E. The tuples in $\mathcal{T}_{\text{rep}}$, which resemble the germs of gate set tomography [62], are parameterised by repetition numbers describing how many times the elements of the tuple are repeated. While we could optimise our choice of repeated tuples, we do not do so here.

We optimise these repetition numbers with a heuristic gradient-free coordinate descent algorithm [106], which cycles through the repetition numbers and steps them towards smaller figures of merit. The algorithm grows the step size exponentially when repeatedly stepping in the same direction, and optimises the shot weights Γ with the gradient descent routine in Section 2.4.2 before evaluating the figure of merit. In practice, it appears most performant to choose repeated tuples that, up to a Pauli correction, implement an involution, so that repeating the tuple twice implements the identity, and use only odd repetition numbers.

After optimising the repetition numbers of the deep tuples, we optimise the shallow tuples with Algorithm 2, a greedy algorithm that resembles the experimental design optimisation algorithm of [107]. It performs a number of *excursions* n_{ex}, each of which greedily grows and shrinks the tuple set. To grow the tuple set, an excursion generates random tuples according to a probability distribution over tuples $\mathcal{P}$ and greedily adds them to the tuple set according to the figure of merit. It does this until the tuple set contains $l_{\text{set}} + l_{\text{ex}}$ tuples, where l_{set} is the desired tuple set size and l_{ex} is the excursion length, by trialling up to f_{trial} times more tuples than it is trying to add. Then to shrink the tuple set, it greedily prunes tuples until either it can no longer remove a tuple without increasing the figure of merit, or the tuple set contains l_{set} tuples.

Although Algorithm 2 could in principle remove deep tuples, or add tuples of intermediate depth, it tends not to do so in practice. We usually choose $n_{\text{ex}} = 3$, $l_{\text{ex}} = 10$, $l_{\text{set}} = 5|\mathcal{I}|$, and $f_{\text{trial}} = 20$, and describe $\mathcal{P}$ in Section 2.E.

Algorithm 2: Tuple set optimisation algorithm

Input: tuple set $\mathcal{T}$, circuit $\mathcal{C}$, gate eigenvalues $\boldsymbol{\lambda}$
Output: tuple set $\mathcal{T}$
Parameters: Excursion number n_{ex} and length l_{ex}, tuple set size l_{set}, trial factor f_{trial}, tuple probability distribution $\mathcal{P}$

for ex **in** $[n_{\text{ex}}]$ **do**
  /* Greedily add tuples to the set */
  initialise $t \leftarrow f_{\text{trial}}\big(l_{\text{set}} + l_{\text{ex}} - |\mathcal{T}|\big)$
  while $|\mathcal{T}| < l_{\text{set}} + l_{\text{ex}}$ **and** $t > 0$ **do**
    sample $T \sim \mathcal{P}$
    if $\mathcal{F}(\mathcal{T} \cup T|\mathcal{C}, \boldsymbol{\lambda}) < \mathcal{F}(\mathcal{T}|\mathcal{C}, \boldsymbol{\lambda})$ **then**
      append $\mathcal{T} \leftarrow \mathcal{T} \cup T$
    decrement $t \leftarrow t - 1$
  /* Greedily remove tuples from the set */
  while $\mathcal{F}(\mathcal{T} \setminus T'|\mathcal{C}, \boldsymbol{\lambda}) < \mathcal{F}(\mathcal{T}|\mathcal{C}, \boldsymbol{\lambda})$ **or** $|\mathcal{T}| > l_{\text{set}}$ **do**
    set $T' \leftarrow \arg\min_{T \in \mathcal{T}} \mathcal{F}(\mathcal{T} \setminus T|\mathcal{C}, \boldsymbol{\lambda})$
    remove $\mathcal{T} \leftarrow \mathcal{T} \setminus T'$
return $\mathcal{T}$

After optimising the tuple set $\mathcal{T}$ with these algorithms, we might in practice add repeated tuples of an intermediate depth to verify the expected exponential decay in the circuit eigenvalues as a function of circuit depth at the cost of performance. Finally, we optimise the shot weights Γ to obtain an optimised experimental design $(\mathcal{T}, \Gamma)$.

2.5 Numerical results

We now present numerical results demonstrating that our methods are scalable and capable of performant noise characterisation of surface code syndrome extraction circuit. These results were produced on a 2021 M1 Max Macbook Pro with 32 GB of RAM with Julia [108], and stabiliser circuits were stimulated with Stim [56]. We release all of our code as the Julia package QuantumACES [88].

We test our methods on a distribution over noise models that we call *log-normal Pauli noise*, where each Pauli error probability associated with each gate is independently log-normally distributed, as this resembles the noise observed in the quantum device featured in the surface code experiment [11]. We choose physically relevant average single-qubit gate, two-qubit gate, and measurement error rates of $r_1 = 0.075\%$, $r_2 = 0.5\%$, and $r_m = 2\%$, respectively, and supply the full details of the noise model, including layer times, in Section 2.F. We will test our noise characterisation methods on the seed-0 random instance of this noise model, and optimise our experimental designs for depolarising noise with the same error rates, where under depolarising noise gates have an equal probability of each Pauli error.

We first optimise an experimental design for depolarising noise on the syndrome extraction circuit of a distance-3 surface code on 17 qubits, using the procedures outlined

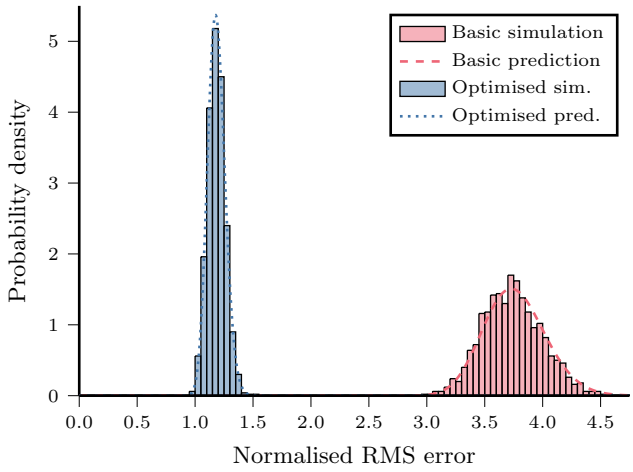

Figure 2.3: Simulated and predicted performance of basic and optimised ACES experimental designs. Simulated data show the distribution of the normalised RMS error between the estimated and true gate eigenvalues when characterising a fixed-seed random instance of log-normal Pauli noise for the syndrome extraction circuit of a distance-3 surface code. The data are 1000 trials of characterising a fixed-seed random instance of log-normal Pauli noise with a measurement budget $S = 10^8$ and align with predicted performance distributions based on Equation 2.27. The optimised experimental design substantially outperforms the basic experimental design.

in Section 2.4. This depolarising noise model functions only as an optimisation target and is constructed from estimates of the average single-qubit gate, two-qubit gate, and measurement error rates, in this case as detailed above for log-normal Pauli noise. The random tuples added by Algorithm 2 are shallow, with depth at most 4, and the shot weights allocate the majority of the measurement budget to individually measuring the controlled-Z gate layers. We present the complete experimental design in Section 2.G.

Then we perform ACES noise characterisation on a fixed-seed random instance of log-normal Pauli noise, comparing the basic and optimised experimental designs. Figure 2.3 shows the distribution of the normalised RMS error over 1000 simulated trials of ACES with a measurement budget $S = 10^8$ for both designs, as well as the predicted normalised RMS error distributions. The predicted distributions closely align with the simulated data, validating our ability to predict the performance of ACES noise characterisation experiments, and demonstrating that the optimised design substantially outperforms the basic design. For this random instance of log-normal Pauli noise, the basic design has a figure of merit, or expected normalised RMS error, that is larger than the optimised design by a factor 3.17, implying that the sample efficiency of the optimised design is a factor 10.1 better than the basic design.

Next, we examine how the performance of this optimised experimental design for the syndrome extraction circuit varies as a function of the surface code distance. Figure 2.4 depicts the figure of merit $\mathcal{F}$, the expected normalised RMS error, and the normalised RMS error standard deviation $\sqrt{\mathcal{V}}$ as functions of the surface code distance d. The exact quantities are reported for depolarising noise, whereas for log-normal Pauli noise, we report estimates of the mean across random samples from the distribution over noise models.

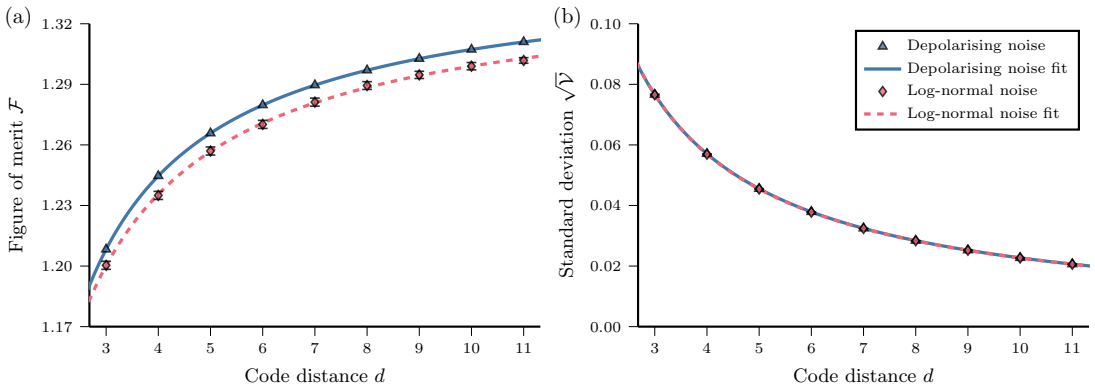

Figure 2.4: Performance scaling of ACES noise characterisation as a function of the surface code distance d. (a) Figure of merit $\mathcal{F}$, the expected normalised RMS error, and (b) the normalised RMS error standard deviation $\sqrt{\mathcal{V}}$, of the optimised experimental design for the syndrome extraction circuit. The data points have no error for depolarising noise, whereas for log-normal Pauli noise, they report the mean estimated by sampling from the distribution over noise models, with error bars indicating two standard deviations. The functional forms, based on Equations 2.28 and 2.29 and discussed in the text, fit the data precisely, with relative errors of under 10^{-8} for each depolarising noise data point.

Empirically, we find that for depolarising noise, the normalised trace of the gate eigenvalue estimator covariance matrix $\mathrm{tr}\,(\Sigma)/S'$, and of its square $\mathrm{tr}\,(\Sigma^2)/S'^2$, are precisely fit as quadratic functions of d. Model selection using the Akaike information criterion corrected for small samples [109] prefers this quadratic model over other polynomial models. Similarly, the number of gate eigenvalues is exactly described by a quadratic with integer coefficients, $N(d) = 84d^2 - 36d - 24$. Substituting these three quadratics into Equations 2.28 and 2.29 yields functional forms that precisely describe the scaling of $\mathcal{F}$ and $\sqrt{\mathcal{V}}$ with d, which are also depicted in Figure 2.4. For depolarising noise, we fit the normalised traces of the gate eigenvalue estimator covariance matrix, obtaining functional forms for $\mathcal{F}$ and $\sqrt{\mathcal{V}}$ with relative errors of less than 10^{-8} for each data point. By contrast, for log-normal Pauli noise, we directly and simultaneously fit the functional forms for $\mathcal{F}$ and $\sqrt{\mathcal{V}}$. Although the data represent estimates, we nevertheless obtain relative errors of less than 10^{-3} for each data point.

Importantly, the functional forms derived by substituting the quadratics into Equations 2.28 and 2.29 entail that in the limit of large d, the figure of merit $\mathcal{F}$, or expected normalised RMS error, approaches a constant value. Also, the optimised experimental design requires 261 experiments, before Pauli frame randomisation, to estimate all of the circuit eigenvalues at all tested code distances d, which range from $d = 3$ to $d = 25$. This demonstrates that ACES is capable of estimating noise in surface code syndrome extraction circuits to a precision that is asymptotically independent of the number of qubits n in the code, using a number of experiments that is also asymptotically independent of n.

Notably, the optimised experimental design performs better on average for log-normal Pauli noise, despite being optimised for depolarising noise. Moreover, Figure 2.5 shows histograms of the figure of merit $\mathcal{F}$ across random instances of log-normal Pauli noise at a range of surface code distances d. It shows that random samples of log-normal Pauli noise have increasingly similar figures of merit as the surface code distance d increases.

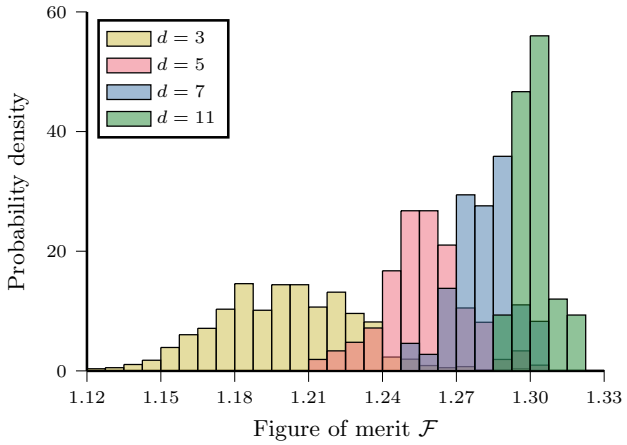

Figure 2.5: Variation in the performance of ACES noise characterisation across random instances of log-normal Pauli noise at a range of surface code distances d. The histograms indicate the distribution of the figure of merit $\mathcal{F}$, the expected normalised RMS error, of the optimised experimental design for the syndrome extraction circuit. Random instances of log-normal Pauli noise have increasingly similar figures of merit with increasing surface code distance d.

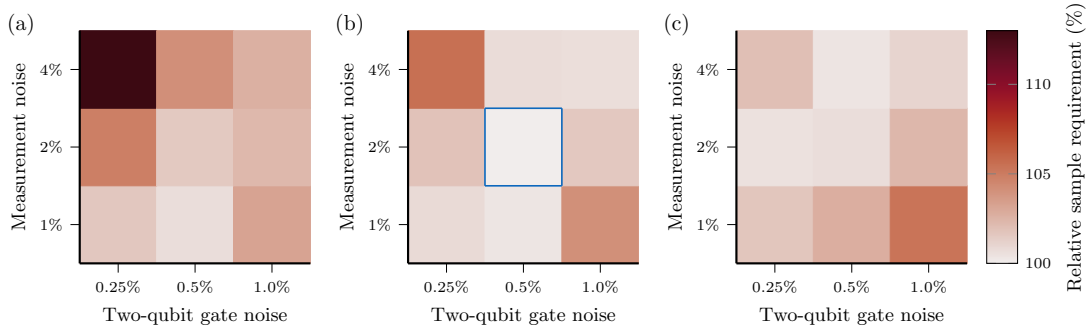

Figure 2.6: Performance of ACES experimental designs optimised for inaccurate average error rates. The experimental designs were optimised for depolarising noise with a range of error rates, with single-qubit gate error rates (a) $r_1 = 0.0375\%$, (b) $r_1 = 0.075\%$, and (c) $r_1 = 0.15\%$, and two-qubit gate and measurement error rates indicated in the heatmaps. Experimental designs were evaluated for the syndrome extraction circuit of a distance-3 surface code against 400 random instances of log-normal Pauli noise, whose average error rates are indicated by the blue square. Heatmap colour indicates the expected number of samples required to achieve a fixed estimation accuracy, expressed relative to the best-performing design optimised at the correct average error rates. Only a single experimental design requires over 10% more samples to achieve the same accuracy as the best-performing design.

Together, these suggest that the performance of the optimised experimental design is not substantially harmed by small changes in the noise model that preserve average error rates. The effects of randomness in the noise model on the performance of ACES also appear to average out with increasing system size.

We also examine experimental designs optimised at different error rates to determine the robustness of their performance. Specifically, we optimise for half the error rate, the error rate, and double the error rate, on single-qubit gates, two-qubit gates, and measurements, yielding 27 different experimental designs. Figure 2.6 shows the expected number of samples required to achieve a fixed estimation accuracy, evaluated over 400 random instances of log-normal Pauli noise for the syndrome extraction circuit of a distance-3 surface code. These values are expressed relative to the original experimental design optimised for the appropriate average error rates, which performs best.

Only 3 of the 27 experimental designs require over 5% more samples than the most performant experimental design to achieve the same expected accuracy, and the only one to require greater than 10% more samples specifically requires $13.00 \pm 0.10\%$. This corresponds to having a mean figure of merit of 1.2757 ± 0.0016, whereas the mean figure of merit of the most performant design is 1.2001 ± 0.0014. The expected relative number of samples required to achieve the same accuracy is the square of this ratio, and is estimated precisely due to substantial covariance in the figure of merit between designs for each instance of log-normal Pauli noise. Overall, this demonstrates that the performance of optimised experimental designs is robust to being optimised for depolarising noise with inappropriate average error rates.

Finally, we use our optimised experimental design to characterise a fixed-seed random instance of log-normal Pauli noise on the syndrome extraction circuit of a distance $d = 25$ surface code with $n = 1249$ qubits. The optimised experimental design features 31 tuples

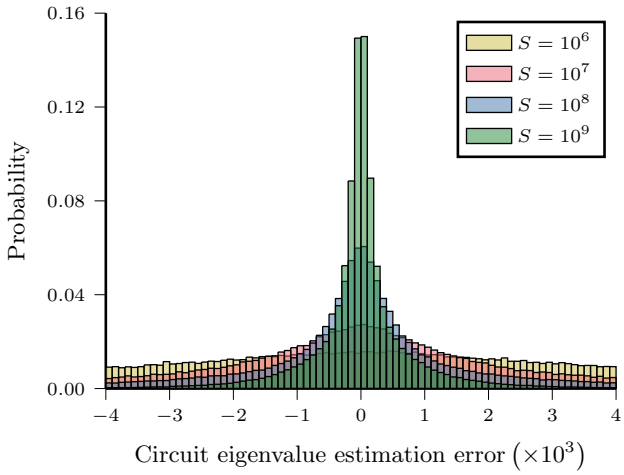

Figure 2.7: Performance of ACES noise characterisation with the optimised experimental design for the syndrome extraction circuit of a distance-25 surface code. The histograms indicate distributions of the circuit eigenvalue estimation error, the difference between the estimated and true circuit eigenvalues. The data are for a fixed-seed random instance of log-normal Pauli noise over a range of measurement budgets S. Circuit eigenvalues are estimated to greater precision with increasing S.

which together estimate 267,357 circuit eigenvalues over 261 experiments, before Pauli frame randomisation, in order to estimate the 51,576 gate eigenvalues. Optimising the experimental design at $d = 3$, generating it at $d = 25$, and then simulating ACES noise characterisation experiments for measurement budgets $S \in \{10^6, 10^7, 10^8, 10^9\}$ together took under 5 hours, with roughly half of that time being dedicated to stabiliser circuit simulations with Stim. The fits in Figure 2.4 predict an average figure of merit 1.3245 and a standard deviation 0.0091, across random instances of log-normal Pauli noise. By comparison, for the simulated noise characterisation of a specific random instance of log-normal Pauli noise, the normalised RMS errors were $\{1.4480, 1.3243, 1.3303, 1.3245\}$ across measurement budgets. Excepting the smallest measurement budget $S = 10^6$, these results are highly consistent with our performance predictions.

The noise estimation procedure begins by estimating the circuit eigenvalues from raw measurement data. Figure 2.7 depicts histograms of the circuit eigenvalue estimation error, the difference between the estimated and true circuit eigenvalues, across the range of measurement budgets. As the measurement budget increases, the distributions of the circuit eigenvalue estimation error narrow, straightforwardly improving the accuracy of the circuit eigenvalue estimates.

Ultimately, the protocol estimates the Pauli error probability distributions of all gates and measurements appearing in the circuit. We measure the gate estimation error with the total variation distance (TVD) between the estimated and true Pauli error probability distribution, a principled measure for probability distributions and equivalent to the diamond norm between Pauli channels [94]. Figure 2.8 depicts histograms of the gate estimation error across measurement budgets and the different types of gate appearing in the circuit, namely: dynamical decoupling X gates and padded identity gates, which are Pauli gates; Hadamard gates; measurements; and controlled-Z gates. The Pauli gates tend to be estimated to a higher precision than the Hadamard gates, despite both being single qubit gates. This is potentially related to the fact that Pauli gates commute with other Pauli gates, up to sign, a setting in which it is easier to achieve relative precision noise estimation, as discussed in Section 2.C. By contrast, the two-qubit controlled-Z gates are least accurately estimated, accounted for in part by their Pauli error probability distribution being over 16 errors, compared to 4 for the single-qubit gates and just 2 for measurements.

The gate estimation errors improve roughly by a constant factor of $\sqrt{10}$ for each factor of 10 increase in the measurement budget S, demonstrating the expected $1/\sqrt{S}$ sample efficiency. This is clear visually and quantified in Table 2.1, which shows the median gate estimation error across gate types and measurement budgets. The Hadamard and controlled-Z gate estimates outperform the trend, particularly for the smallest measurement budget $S = 10^6$. This is because large estimation errors for small values of S are improved by ensuring parameter estimates are within bounds. Specifically, gate eigenvalues greater than 1 are set to 1, and estimated probability distributions are projected into the simplex. For the data in Table 2.1, at $S = 10^6$, 1211 gate eigenvalues are set to 1, while at $S = 10^7$, only 51 gate eigenvalues are set to 1.

By contrast, referring again to Table 2.1, we see that measurement error estimation underperforms the trend at $S = 10^6$. This appears to be the cause of the outlying poor normalised RMS error 1.4480 at this measurement budget. Our performance prediction calculations implicitly assume that we are in the limit of collecting a large number of samples, so poor prediction at the smallest measurement budget is not surprising. Nevertheless, performance is highly consistent with our predictions at all larger measurement

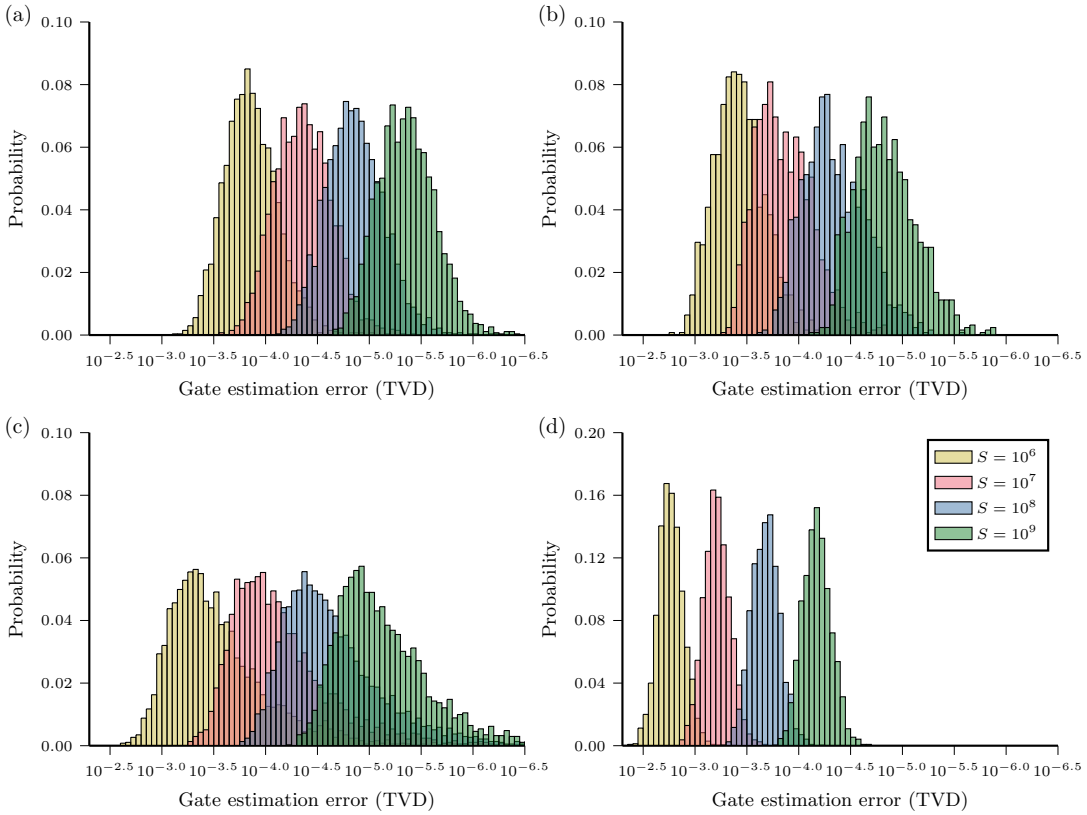

Figure 2.8: Performance of ACES noise characterisation with the optimised experimental design for the syndrome extraction circuit of a distance-25 surface code. The histograms indicate distributions of the gate estimation error, the total variation distance (TVD) between the estimated and true error probability distributions for the gate. The data are for a fixed-seed random instance of log-normal Pauli noise over a range of measurement budgets S. The estimation error distributions are shown across the range of gate types appearing in the circuit: (a) dynamical decoupling X gates and padded identity gates, or Pauli gates; (b) Hadamard gates; (c) measurements; and (d) controlled-Z gates. Across all gate types, the estimation error distributions consistently shift roughly by a constant factor of $1/\sqrt{10}$ when the measurement budget S is increased by a factor of 10, demonstrating the anticipated sample efficiency.

Table 2.1: Median of the gate estimation error distributions shown in Figure 2.8. Results are shown for the optimised experimental design over a range of measurement budgets S, and across the range of gate types appearing in the circuit: dynamical decoupling X gates and padded identity gates, or Pauli gates; Hadamard gates; measurements; and controlled-Z gates. Sequential entries differ by roughly 0.5, confirming the anticipated $1/\sqrt{S}$ scaling of the estimation error.

Measurement budget	$-\log_{10}$ median gate estimation error (TVD)			
	Pauli	Hadamard	Measurement	Controlled-Z
10^6	3.841	3.415	3.437	2.761
10^7	4.353	3.829	4.045	3.205
10^8	4.850	4.313	4.548	3.680
10^9	5.348	4.805	5.051	4.171

budgets.

We have demonstrated scalable and performant noise characterisation of the syndrome extraction circuits of very large surface codes in a physically-relevant parameter regime. We are able to precisely predict the performance of ACES noise characterisation experiments, including as functions of the surface code distance. Optimising experimental designs for our figure of merit robustly improves performance, even for a different noise model or average error rates. We supply further numerical results in Section 2.G, and reproduce them for the unrotated surface code in Section 2.H.

2.6 Conclusions

In this paper, we have described and numerically validated a scalable noise characterisation protocol based on ACES. Our protocol generates optimised experimental designs, leveraging our precise analysis of the performance of ACES noise characterisation experiments. We show in numerical simulations of a distance-25 surface code with 1249 qubits that we can achieve characterisation of circuit-level Pauli noise with predictable performance. We believe the methods outlined here could be extended to all components of fault-tolerant quantum computation with topological quantum codes.

There are a large number of potential directions for future work. A rigorous and general theoretical understanding of tuple sets and their performance characteristics would be clarifying, and would likely suggest improved optimisation algorithms. In particular, our empirical ability to precisely predict the performance scaling of ACES noise characterisation of syndrome extraction circuits as a function of the surface code distance suggests underlying structure that we have not uncovered here. We may be able to improve the performance of ACES by drawing more heavily on the theory of optimal experimental designs [100], as well as on experimental design techniques used in other noise characterisation methods. Salient additions include optimising our choice of repeated tuples and removing measurements of circuit eigenvalues that do not improve performance, analogous to germ selection and fiducial-pair reduction in gate set tomography [62]. A detailed investigation of the relative precision estimation capabilities of ACES would require marginalising noise estimates across gate Pauli orbits, as in cycle

error reconstruction [76], and may require projecting noise estimates into the probability simplex using the Mahalanobis distance [110], rather than the Euclidean distance, to retain covariance matrix information.

A natural next step would be practically implementing our protocol to characterise noise in a quantum device operating the syndrome extraction circuit of a topological quantum code. Operating a quantum error correcting code entails mid-circuit measurement and reset as part of syndrome extraction, and although we do not characterise the associated noise here, we believe ACES can straightforwardly be modified to characterise the associated noise, following methods such as those in [111, 112]. The resulting noise estimates could then be used for error mitigation, numerical simulations, or supplied to a noise-aware decoder to determine the extent to which detailed noise characterisation can improve code performance.

Acknowledgements

We thank Steven T. Flammia for discussions. ETH is supported by an Australian Government Research Training Program Scholarship. RH is supported by the Sydney Quantum Academy. This work was supported by the Australian Research Council Centre of Excellence for Engineered Quantum Systems (CE170100009) and the U.S. Army Research Office (W911NF-21-1-0001).

2.A ACES estimates marginal Pauli error probabilities

When presenting ACES in Section 2.3.2, we assumed that each layer $\mathcal{C}_i$ of a circuit $\mathcal{C}$ acting on n qubits is associated with an n-qubit Pauli noise channel $\mathcal{E}_i$. As it is generally intractable to learn all 4^n eigenvalues of n-qubit Pauli channels, we focus on learning a circuit-level noise model. Each layer $\mathcal{C}_i$ is decomposed into gates $\mathcal{G}_{ij}$ supported on mutually disjoint sets of qubits, and we learn Pauli channels $\mathcal{E}_{ij}$ associated with the gates $\mathcal{G}_{ij}$.

When estimating the eigenvalues of the gate Pauli channel $\mathcal{E}_{ij}$, we in fact estimate a subset of the eigenvalues of the layer Pauli channel $\mathcal{E}_i$, namely those eigenvalues corresponding to Paulis supported on the gate $\mathcal{G}_{ij}$. It is a consequence of Lemma 4, and by extension Equations 36 and 37, of [35], that the corresponding Pauli error probabilities of the gate channel $\mathcal{E}_{ij}$ are the marginal of the Pauli error probability distribution of the layer channel $\mathcal{E}_i$ onto the support of the gate $\mathcal{G}_{ij}$. Hence when ACES estimates the noise channel $\mathcal{E}_{ij}$ of the gate $\mathcal{G}_{ij}$ in the context of the layer $\mathcal{C}_i$, it captures the averaged effect on that particular gate of the spatial correlations in the full n-qubit noise channel.

We now prove this claim directly, using the Pauli transfer matrix and superoperator formalism described in, for example, [113], but omitting much of the formalism of [35]. Write $\boldsymbol{\lambda}$ to refer to the vector of eigenvalues of $\mathcal{E}_i$, namely $\lambda_{\boldsymbol{a}}$ for $\boldsymbol{a} \in \mathbf{P}^n$, and similarly write the vector of Pauli error probabilities as $\boldsymbol{p}$. As explained in Equations 2.7 and 2.8, these are related as $\boldsymbol{\lambda} = W\boldsymbol{p}$ by a Walsh-Hadamard transform matrix W ordered by the symplectic form ω, given as

$$W = \sum_{\boldsymbol{a},\boldsymbol{b} \in \mathbf{P}^n} (-1)^{\omega(\boldsymbol{a},\boldsymbol{b})} |\boldsymbol{a}\rangle\rangle\langle\langle \boldsymbol{b}|. \tag{2.32}$$

This matrix is its own inverse up to a constant factor of 4^{-n}, as $W^2 = 4^n I$, allowing us to easily estimate $\boldsymbol{p}$ from $\boldsymbol{\lambda}$.

Consider a subgroup $A \subseteq \mathbf{P}^n$, and write $\boldsymbol{\lambda}_A$ to refer to the eigenvalues $\lambda_{\boldsymbol{a}}$ for $\boldsymbol{a} \in A$. These are related as $\boldsymbol{\lambda}_A = \Pi_A \boldsymbol{\lambda}$ by the projector onto A, which is simply

$$\Pi_A = \sum_{\boldsymbol{a} \in A} |\boldsymbol{a}\rangle\rangle\langle\langle \boldsymbol{a}|. \tag{2.33}$$

We seek to determine the properties of the Pauli error probability distribution estimated from $\boldsymbol{\lambda}_A$. To avoid introducing additional formalism from [35], we specialise our analysis to the subgroup $A = \text{Pauli}(\mathcal{G}_{ij})$ of Paulis supported on some gate $\mathcal{G}_{ij}$ occurring in the layer $\mathcal{C}_i$. This does not place any restrictions on the support of $\mathcal{G}_{ij}$, which we write as $S_{ij} = \text{supp}(\mathcal{G}_{ij}) \subseteq [n]$, an arbitrary subset of the qubits. In this case, the eigenvalues $\boldsymbol{\lambda}_A$ are precisely the eigenvalues of $\mathcal{E}_i$ we measure in the process of estimating $\mathcal{E}_{ij}$.

Instead denoting $A = \text{Pauli}(\mathcal{G}_{ij}) = \text{Pauli}(S_{ij})$ allows us to write $A^C = \text{Pauli}(S_{ij}^C) = \text{Pauli}([n] \setminus S_{ij})$ as the subgroup of Paulis supported on the set complement $S_{ij}^C = [n] \setminus S_{ij}$. Then, we can clearly write any Pauli $\boldsymbol{a} \in \mathbf{P}^n$ as a sum $\boldsymbol{a} = \boldsymbol{b} + \boldsymbol{c}$ for some $\boldsymbol{b} \in A$ and $\boldsymbol{c} \in A^C$. Indeed, for any $\boldsymbol{b} \in A$ and $\boldsymbol{c} \in A^C$, $\omega(\boldsymbol{b}, \boldsymbol{c}) = 0$ as Paulis supported on mutually disjoint sets of qubits trivially commute. In the language of [35], this is because A^C is the commutant of A and A is its own anticommutant.

This allows us to rewrite Equation 2.7 for eigenvalues $\lambda_{\boldsymbol{a}}$ such that $\boldsymbol{a} \in A$, which

yields

$$\begin{aligned}\lambda_{\boldsymbol{a}} &= \sum_{\boldsymbol{b}\in A}\sum_{\boldsymbol{c}\in A^C}(-1)^{\omega(\boldsymbol{a},\boldsymbol{b}+\boldsymbol{c})}p_{\boldsymbol{b}+\boldsymbol{c}} \\ &= \sum_{\boldsymbol{b}\in A}(-1)^{\omega(\boldsymbol{a},\boldsymbol{b})}\left(\sum_{\boldsymbol{c}\in A^C}p_{\boldsymbol{b}+\boldsymbol{c}}\right).\end{aligned} \tag{2.34}$$

That is, the eigenvalues $\boldsymbol{\lambda}_A$ depend only on the distribution $\boldsymbol{p}$ marginalised over the Paulis in A^C onto the Paulis in A. We can write this as $\boldsymbol{p}_A$, being careful to distinguish the meaning from $\boldsymbol{\lambda}_A$, which refers to the eigenvalues for the Paulis in A.

Indeed, the eigenvalues $\boldsymbol{\lambda}_A$ are related to the marginal distribution $\boldsymbol{p}_A$ by the projection of the Walsh-Hadamard transform onto A, namely $W_A = \Pi_A W \Pi_A$. Calculating, we see that

$$\begin{aligned}W_A^2 &= \sum_{\boldsymbol{a},\boldsymbol{a}',\boldsymbol{b},\boldsymbol{b}'\in A}(-1)^{\omega(\boldsymbol{a},\boldsymbol{b})+\omega(\boldsymbol{a}',\boldsymbol{b}')}|\boldsymbol{b}'\rangle\rangle\langle\langle\boldsymbol{a}'|\boldsymbol{a}\rangle\rangle\langle\langle\boldsymbol{b}| \\ &= \sum_{\boldsymbol{b},\boldsymbol{b}'\in A}\left(\sum_{\boldsymbol{a}\in A}(-1)^{\omega(\boldsymbol{a},\boldsymbol{b}+\boldsymbol{b}')}\right)|\boldsymbol{b}'\rangle\rangle\langle\langle\boldsymbol{b}| \\ &= |A|\sum_{\boldsymbol{b}\in A}|\boldsymbol{b}\rangle\rangle\langle\langle\boldsymbol{b}| = |A|\Pi_A.\end{aligned} \tag{2.35}$$

The third line follows from Lemma 1 in [35], the fact that any Pauli commutes either with all Paulis in a subgroup of $\mathbf{P}^n$, or exactly half of them. Hence W_A behaves similarly to W in the sense that W_A squares to Π_A, up to a constant, whereas W squares to the identity, the projector onto the entirety of $\mathbf{P}^n$.

We have shown that when we attempt to estimate the eigenvalues of the gate Pauli channel $\mathcal{E}_{ij}$ associated with the gate $\mathcal{G}_{ij}$, we in fact estimate some of the eigenvalues of the layer Pauli channel $\mathcal{E}_i$ associated with the layer $\mathcal{C}_i$ in which $\mathcal{G}_{ij}$ appears. Taking the Walsh-Hadamard transform of these eigenvalues yields estimates for the Pauli error probabilities of $\mathcal{E}_{ij}$ that correspond to the Pauli error probability distribution of $\mathcal{E}_i$ marginalised onto the support of the gate $\mathcal{G}_{ij}$. This gives a sense in which marginalisation commutes with the Walsh-Hadamard transform.

Consequently, the ACES estimate of the noise associated with a particular gate operated in the context of a particular layer captures the averaged effect of spatial correlations within the layer on that gate. Padding layers with single-qubit identity gates on qubits not included in the support of some existing gate in the layer ensures that we learn marginals of the full Pauli error probability distribution onto all qubits.

2.B Least squares estimation methods

In Section 2.3.4, we described a weighted least squares (WLS) method for solving the standard linear regression problem of Equation 2.24. Alternatives include ordinary least squares (OLS) and generalised least squares (GLS). In this appendix, we describe and compare these methods and explain our preference for WLS, drawing on standard linear regression theory described, for example, in [114].

Recall Equation 2.24, which gives the standard form linear regression problem

$$\boldsymbol{b} = A\boldsymbol{x} + \boldsymbol{\varepsilon}, \tag{2.36}$$

where the circuit log-eigenvalues $\boldsymbol{b}$ are related to the gate log-eigenvalues $\boldsymbol{x}$ by the design matrix A, alongside an error variable we assume is distributed according to a multivariate normal distribution $\boldsymbol{\varepsilon} \sim \mathcal{N}_M(0, \Omega')$. The least squares estimators for the gate log-eigenvalues in the linear regression problem Equation 2.24 can all be expressed in the form

$$\hat{\boldsymbol{x}} = \left(A^\intercal \hat{P} A\right)^{-1} A^\intercal \hat{P} \boldsymbol{b}, \tag{2.37}$$

where the nature of the estimated precision matrix $\hat{P}$ differentiates OLS, WLS, and GLS.

OLS simply sets the precision matrix to be the identity, $\hat{P} = I$. This corresponds to assuming that the error term $\boldsymbol{\varepsilon}$ is homoscedastic, that is, ε_μ has the same variance for each $\mu \in M$, and uncorrelated, that is, there is zero covariance between ε_α and ε_β for all $\alpha \neq \beta$. These assumptions entail that the covariance matrix of $\boldsymbol{\varepsilon}$ is proportional to the identity. WLS relaxes the assumption of homoscedasticity, taking $\hat{P}_{\mu\mu} = \hat{\Omega}'^{-1}_{\mu\mu}$ to be a diagonal matrix whose entries are the inverse estimates of the circuit log-eigenvalue variances. GLS relaxes both assumptions, taking $\hat{P} = \hat{\Omega}'^{-1}$ to be the inverse of the estimated circuit log-eigenvalue covariance matrix.

The accuracy of the least squares estimator improves as we progress from OLS to WLS and then to GLS, each time using a more detailed precision matrix. Note also that GLS simplifies the expression for the gate log-eigenvalue estimator covariance matrix in Equation 2.26 to $\Sigma' = \left(A^\intercal \Omega'^{-1} A\right)^{-1}$, under the assumption $\hat{\Omega}' = \Omega'$ we use generally when calculating the figure of merit.

Implementing WLS and GLS requires us to estimate the diagonal elements of the circuit log-eigenvalue covariance matrix, and the full matrix Ω', respectively. Recall that the entries of Ω' are given by Equation 2.23. We can estimate the variance of the circuit eigenvalue estimator $\hat{\Lambda}_{T,\boldsymbol{a}}$ using that estimate for the circuit eigenvalue, making WLS easy to implement. To construct the full covariance matrix, we may also need to calculate the off-diagonal elements $\Lambda_{T,\boldsymbol{a}+\boldsymbol{a}'}$ from the gate eigenvalues using Equation 2.18. Write A_μ to denote the vector whose elements are $A_{\mu\nu}$, the powers of the gate eigenvalues appearing in Equation 2.18 for the circuit eigenvalue Λ_μ. This decomposition shows that the covariance will be zero when $A_{T,\boldsymbol{a}} + A_{T,\boldsymbol{a}'} = A_{T,\boldsymbol{a}+\boldsymbol{a}'}$, and strictly positive otherwise. The former condition is clearly satisfied when the supports of $P_{\boldsymbol{a}}$ and $P_{\boldsymbol{a}'}$ remain disjoint throughout the action of the circuit $\mathcal{C}_T$, removing the need to calculate $A_{T,\boldsymbol{a}+\boldsymbol{a}'}$ in that case. It also implies that the covariance matrix is sparse even within the blocks for each tuple $T \in \mathcal{T}$.

In this case, we cannot directly implement GLS, as the gate eigenvalues are what we are attempting to estimate. Instead, we perform feasible generalised least squares (FGLS), first estimating the gate eigenvalues with WLS, and then iteratively generating Ω' from the estimates and performing GLS until convergence. In practice, FGLS performs similarly to GLS with the true covariance matrix.

To efficiently numerically implement GLS, we take the sparse Cholesky factorisation $\hat{\Omega}' = LL^\intercal$ and then left-multiply the linear regression problem Equation 2.24 by L^{-1}. Indeed, as Ω' is block diagonal, we can perform this calculation separately for the blocks of Ω', which correspond to the tuples in the tuple set.

Even so, the problems of calculating the sparse Cholesky factorisation of $\hat{\Omega}'$ and inverting the Cholesky factor L are intractable for very large surface codes. For example, at $d = 25$ there are hundreds of thousands of circuit eigenvalues. We therefore focus on WLS in this paper, as we are primarily interested in detailing a practical and scalable protocol. Nevertheless, GLS offers greater performance for small-scale noise

characterisation experiments and should be used in these instances. The results throughout this paper are similar for GLS.

2.C ACES estimates noise to relative precision

In this appendix, we optimise an experimental design for a toy circuit to demonstrate that ACES is capable of learning Pauli channels to relative precision, including in the presence of SPAM noise. We achieve this with the usual strategy employed in tomography and noise characterisation, namely, by using a tuple that amplifies noise by repeating the layer a large number of times.

It is not generally possible to estimate all eigenvalues of a Pauli noise channel associated with a Clifford gate to relative precision [115]. In particular, when single-qubit gates have no appreciable noise, we find that ACES is capable of relative precision estimation of the eigenvalues of those Paulis whose support is unchanged by the Clifford gate, in line with [115]. By contrast, when single-qubit gates are appreciably noisy, we find that ACES is only capable of relative precision estimation of the eigenvalues of Paulis that commute with the Clifford gate up to sign. This is because only the products of gate eigenvalues within each of the orbits of the gate can be learned to relative precision, as discussed in cycle error reconstruction [76]. We sidestep this issue here by considering only single-qubit Pauli gates that commute with all Paulis up to sign, though we believe that ACES is generally capable of relative precision noise estimation.

However, we caveat that this appendix demonstrates that full relative precision estimation is in fact not possible when accounting for experiment duration: it is only possible when simply considering sample complexity. Accounting for the time taken to implement experiments penalises the deep circuits used in relative precision noise estimation. This results in circuits of more moderate depth—which are not capable of full relative precision noise estimation—becoming optimal.

Consider a circuit acting on n qubits that consists of a single layer of n single-qubit Pauli gates, with SPAM noise. Each gate has 3 gate eigenvalues whose values we all set to λ, and each of the 3 Pauli preparation and measurements on each qubit is assigned a SPAM noise eigenvalue λ_m, so that overall we have $N = 6n$ gate eigenvalues.

We specify our tuple set as consisting of two tuples that repeat the layer ϕ_1 and ϕ_2 times, respectively. Our experimental design is parameterised by the layer repetition numbers ϕ_1 and ϕ_2, and without loss of generality we take $\phi_2 > \phi_1$ so that we can write $\phi_2 = \phi_1 + \phi$. Note that $\phi_1 = 0$ and $\phi_2 = 1$ corresponds to the basic tuple set. Ordering the circuit and gate eigenvalues appropriately yields the design matrix

$$A = \begin{bmatrix} \phi_1 I_{3n} & I_{3n} \\ \phi_2 I_{3n} & I_{3n} \end{bmatrix}. \tag{2.38}$$

Next, we need to determine the circuit eigenvalue estimator covariance matrix. To measure all the circuit eigenvalues, each experiment merely needs to prepare and measure each of Pauli X, Y and Z on the n qubits, resulting in $|\mathcal{E}_T| = 3$ experiments for any tuple T. This scheme implies that none of the circuit eigenvalue estimators are correlated with each other, producing a diagonal covariance matrix. Now suppose that measurement and reset takes a time τ relative to the circuit layer. The time factor for the basic tuple set is

$$\tau_{\mathcal{T}_\mathcal{I}} = \frac{2\tau(\tau+1)}{2\tau+1}. \tag{2.39}$$

Our experimental design assigns a shot weight Γ to the ϕ_2 tuple, and $1-\Gamma$ to the ϕ_1 tuple, yielding a time factor

$$\tau_{\mathcal{T},\Gamma} = \tau + \phi_1 + \Gamma\phi. \tag{2.40}$$

The measurement budget S is related to the measurement shots S' as

$$S = S'\left(\frac{2\tau(\tau+1)}{(2\tau+1)(\tau+\phi_1+\Gamma\phi)}\right). \tag{2.41}$$

Then, with reference to Equation 2.23 we see that the circuit eigenvalue estimator covariance matrix Ω is

$$\Omega = \frac{3}{S}\begin{bmatrix}\omega_1 I_{3n} & 0\\ 0 & \omega_2 I_{3n}\end{bmatrix}, \tag{2.42}$$

$$\omega_1 = \frac{1-\lambda_m^2\lambda^{2\phi_1}}{1-\Gamma}, \tag{2.43}$$

$$\omega_2 = \frac{1-\lambda_m^2\lambda^{2\phi_2}}{\Gamma}. \tag{2.44}$$

A first-order Taylor expansion yields

$$\Omega' = \frac{3}{S}\begin{bmatrix}\omega_1' I_{3n} & 0\\ 0 & \omega_2' I_{3n}\end{bmatrix}, \tag{2.45}$$

$$\omega_1' = \frac{1-\lambda_m^2\lambda^{2\phi_1}}{(1-\Gamma)\lambda_m^2\lambda^{2\phi_1}}, \tag{2.46}$$

$$\omega_2' = \frac{1-\lambda_m^2\lambda^{2\phi_2}}{\Gamma\lambda_m^2\lambda^{2\phi_2}}. \tag{2.47}$$

As the covariance matrix is diagonal, we can treat WLS and GLS to approximate $\hat{W} = \Omega'^{-1}$, simplifying Equation 2.26 to $\Sigma' = \left(A^\intercal\Omega'^{-1}A\right)^{-1}$. Calculating, and performing another first-order Taylor expansion, we obtain

$$\Sigma' = \frac{3}{S}\begin{bmatrix}\frac{\omega_1'+\omega_2'}{\phi^2}I_{3n} & -\frac{\phi_2\omega_1'+\phi_1\omega_2'}{\phi^2}I_{3n}\\ -\frac{\phi_2\omega_1'+\phi_1\omega_2'}{\phi^2}I_{3n} & \frac{\phi_2^2\omega_1'+\phi_1^2\omega_2'}{\phi^2}I_{3n}\end{bmatrix}, \tag{2.48}$$

$$\Sigma = \frac{3}{S}\begin{bmatrix}\frac{\omega_1'+\omega_2'}{\phi^2}\lambda^2 I_{3n} & -\frac{\phi_2\omega_1'+\phi_1\omega_2'}{\phi^2}\lambda_m\lambda I_{3n}\\ -\frac{\phi_2\omega_1'+\phi_1\omega_2'}{\phi^2}\lambda_m\lambda I_{3n} & \frac{\phi_2^2\omega_1'+\phi_1^2\omega_2'}{\phi^2}\lambda_m^2 I_{3n}\end{bmatrix}. \tag{2.49}$$

The top-left block of Σ describes the variance of the gate eigenvalue estimators, whereas the bottom-right block describes the variance of the measurement eigenvalue estimators, and the remaining blocks describe the covariance between the gate and measurement eigenvalue estimators.

It is simplest and most instructive to focus on the large n limit. Examining Equation 2.28, and noting that both $\mathrm{tr}\,(\Sigma)$ and $\mathrm{tr}\,(\Sigma^2)$ are proportional to n, we see that the figure of merit reduces to $\mathcal{F} = \sqrt{S'/N}\sqrt{\mathrm{tr}\,(\Sigma)}$.

We are interested here in relative precision noise estimation, which is only possible for the gate eigenvalues. We will therefore consider a modified version of the figure of merit, which for convenience we still call $\mathcal{F}$, where the trace is taken only over the top-left block of Σ corresponding to the gate eigenvalues. For this modified figure of merit, we see that

$$\mathcal{F} = \sqrt{\frac{(2\tau+1)(\tau+\phi_1+\Gamma\phi)}{4\tau(\tau+1)}\left(\frac{f_1}{1-\Gamma}+\frac{f_2}{\Gamma}\right)}, \tag{2.50}$$

$$f_1 = \frac{\lambda^2(\lambda^{-2\phi_1}-\lambda_m^2)}{\phi^2}, \tag{2.51}$$

$$f_2 = \frac{\lambda^2(\lambda^{-2\phi_1}\lambda^{-2\phi}-\lambda_m^2)}{\phi^2}. \tag{2.52}$$

Next, let us optimise the shot weight Γ. Inspecting $\mathcal{F}$, the optimal value lies in $(0,1)$ and can be found by setting the derivative with respect to Γ to be zero. This derivative is given by

$$\frac{\partial\mathcal{F}}{\partial\Gamma} = \frac{2\tau+1}{8\tau(\tau+1)\mathcal{F}'}\left(\frac{(\tau+\phi_2)f_1}{(1-\Gamma)^2}-\frac{(\tau+\phi_1)f_2}{\Gamma^2}\right), \tag{2.53}$$

and is zero when

$$\Gamma = \frac{\sqrt{(\tau+\phi_1)f_2}}{\sqrt{(\tau+\phi_2)f_1}+\sqrt{(\tau+\phi_1)f_2}}. \tag{2.54}$$

Substituting and rearranging eventually yields

$$\mathcal{F} = \sqrt{\frac{(2\tau+1)}{4\tau(\tau+1)}}\left(\sqrt{(\tau+\phi_1)f_1}+\sqrt{(\tau+\phi_2)f_2}\right). \tag{2.55}$$

Now, we optimise the repetition number ϕ_1. Differentiating with respect to ϕ_1 yields

$$\begin{aligned}\frac{\partial\mathcal{F}}{\partial\phi_1} = \sqrt{\frac{(2\tau+1)}{4\tau(\tau+1)}}\Bigg(&\frac{\lambda^2\left(\lambda^{-2\phi_1}\left(1-2(\tau+\phi_1)\log(\lambda)\right)-\lambda_m^2\right)}{2\phi^2\sqrt{(\tau+\phi_1)f_1}}\\ &+\frac{\lambda^2\left(\lambda^{-2\phi_2}\left(1-2(\tau+\phi_2)\log(\lambda)\right)-\lambda_m^2\right)}{2\phi^2\sqrt{(\tau+\phi_2)f_2}}\Bigg).\end{aligned} \tag{2.56}$$

Since $\lambda \in (\delta, 1]$ for some $\delta > 0$, we see that $\log(\lambda)$ is non-positive. Inspecting this expression, including f_1 and f_2, which are both always positive, it becomes clear that this derivative is positive for all ϕ_1. This indicates that the optimal value of ϕ_1 is the minimal one, namely 0, as suggested by Elfving's theorem. Then, substituting ϕ_1 into the expression for the figure of merit gives

$$\mathcal{F} = \sqrt{\frac{(2\tau+1)}{4\tau(\tau+1)}}\frac{\lambda}{\phi}\left(\sqrt{\tau(1-\lambda_m^2)}+\sqrt{(\tau+\phi)(\lambda^{-2\phi}-\lambda_m^2)}\right). \tag{2.57}$$

A similar calculation obtains the figure of merit when we do not account for the time taken to perform the gates, which we will call $\mathcal{F}'$, as

$$\mathcal{F}' = \frac{\lambda}{\sqrt{2}\phi}\left(\sqrt{1-\lambda_m^2}+\sqrt{\lambda^{-2\phi}-\lambda_m^2}\right). \tag{2.58}$$

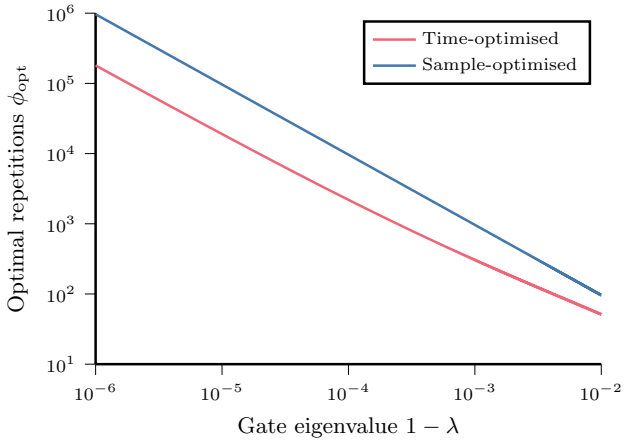

Figure 2.9: The optimal repetition number ϕ_{opt} for estimating gate eigenvalues λ is smaller when we account for gate times than when we do not, which correspond respectively to the time- and sample-optimised experimental designs. Nevertheless, both optimal repetition numbers scale asymptotically as $1/(1-\lambda)$.

Optimising for this figure of merit $\mathcal{F}'$ produces *sample-optimised* experimental designs that maximise estimation precision given a fixed number of samples from the quantum device. By contrast, optimising for the usual figure of merit $\mathcal{F}$ produces *time-optimised* experimental designs that maximise estimation precision given a fixed window of time in which to sample from the quantum device.

Now we are ready to numerically optimise the number of repetitions ϕ. To do this, we optimise it as a continuous variable and then consider both the floor and ceiling of the optimal value, choosing the one that minimises the figure of merit as the optimal repetition number ϕ_{opt}. For these numerical results, we choose parameter values in line

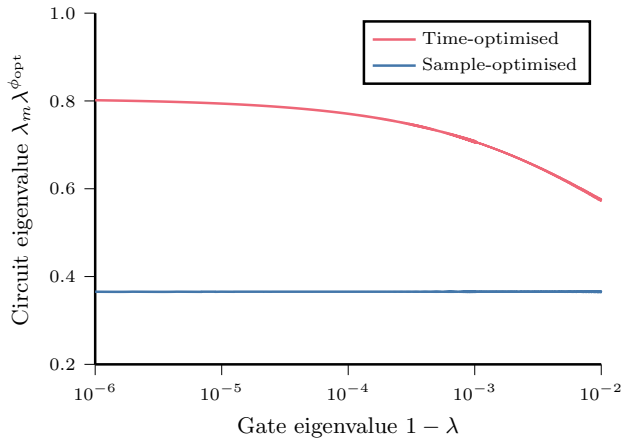

Figure 2.10: The optimal circuit eigenvalue $\lambda_m \lambda^{\phi_{\text{opt}}}$ for estimating gate eigenvalues λ is larger and only slowly approaches a constant value when we account for gate times compared to when we do not. These correspond respectively to the time- and sample-optimised experimental designs.

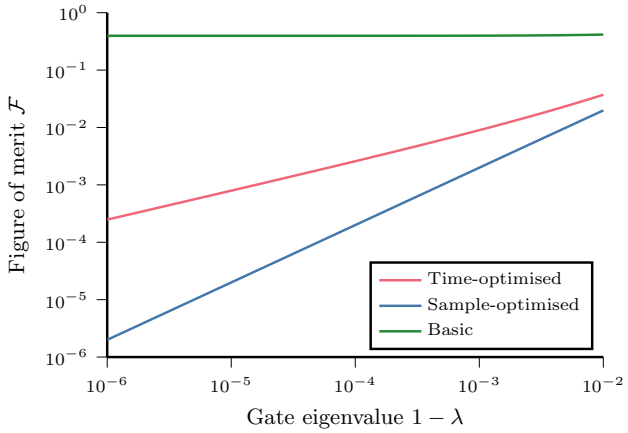

Figure 2.11: The figure of merit $\mathcal{F}$ of ACES noise characterisation experimental designs when estimating gate eigenvalues λ. The basic experimental design is only capable of additive precision, as it does not repeat gates. The figures of merit of the sample- and time-optimised experimental designs scale as $1-\lambda$ and $\sqrt{1-\lambda}$, respectively, demonstrating that ACES is capable of relative precision noise estimation when not accounting for experiment duration.

with the depolarising noise model described in Section 2.F and used elsewhere in this paper. Namely, we choose $\tau = 660/29$ and $\lambda_m = 0.96$, and note that for single-qubit gates we usually have $\lambda_1 = 0.999$, so $1-\lambda_1 = 10^{-3}$. As we focus on relative precision noise estimation, we are primarily interested in the behaviour of quantities as the gate eigenvalue λ approaches 1 from below. We also optimise ϕ for both the usual figure of merit $\mathcal{F}$ which accounts for the time taken to perform gates, as well as $\mathcal{F}'$, which does not.

Figure 2.9 shows the optimal repetition number ϕ_{opt} for estimating gate eigenvalues λ. Asymptotically the optimal number of repetitions scales as $1/(1-\lambda)$, consistent with previous results in randomised benchmarking [116], regardless of whether we account for gate times. Accordingly, we find that the optimal circuit eigenvalue $\lambda_m \lambda^{\phi_{\text{opt}}}$ for estimating gate eigenvalues λ approaches a constant value, as seen in Figure 2.10. The optimal circuit eigenvalue is relatively large and approached slowly when accounting for gate times, whereas when we do not, it is smaller and approximately constant regardless of λ.

Finally, Figure 2.11 shows the figure of merit of a number of experimental designs when estimating gate eigenvalues λ. The figure of merit of the sample-optimised experimental design scales as $(1-\lambda)$, demonstrating that ACES is capable of relative precision noise estimation. By contrast, the basic experimental design, which has $\phi = 1$, has a roughly constant figure of merit and is only capable of additive precision. However, the figure of merit of the time-optimised experimental design scales as $\sqrt{1-\lambda}$.

This toy example elucidates that full relative precision noise estimation, where estimation accuracy scales as $(1-\lambda)$, is not possible when appropriately accounting for the time taken to implement gates on the device. Repeating gates a large number of times proportionally increases the amount of time taken to implement the circuit on a quantum device. This decreases the number of samples collected in a fixed window of time, reducing noise estimation precision. When accounting for this, the accuracy of the optimised experimental design instead scales as $\sqrt{1-\lambda}$.

2.D Figure of merit differentiation

In Section 2.4.2, we used the derivative of the figure of merit $\mathcal{F}$ with respect to the shot log-weights $\gamma = \{\gamma_T\}_{T\in\mathcal{T}}$ associated with a tuple set $\mathcal{T}$ to optimise the shot weights. In this appendix, we detail analytic expressions for this derivative for weighted least squares as well as generalised and ordinary least squares, which are discussed in Section 2.B. This optimisation is crucial, as we find numerical automatic differentiation to be impractically slow. We used an algorithmic tool [117, 118] to obtain analytic expressions for the derivatives presented here, and verified them against automatic differentiation [119].

Recall that the circuit eigenvalue estimator covariance matrix Ω, given by Equation 2.23, is a block diagonal symmetric matrix whose blocks correspond to each tuple T in the tuple set $\mathcal{T}$, weighted by a factor $1/\Gamma_T$, alongside an overall experiment time factor $\tau_{\mathcal{T},\Gamma} = \sum_{T\in\mathcal{T}} \Gamma_T \tau_T$. The block diagonal structure ensures that first-order Taylor expansion used to derive the circuit log-eigenvalue estimator covariance Ω' from Ω commutes with this weighting. Consequently, we work with the more convenient Ω' appearing in Equation 2.26 for the gate log-eigenvalue estimator covariance matrix Σ'. Note also that $\Sigma = \operatorname{diag}(\boldsymbol{\lambda})\Sigma'\operatorname{diag}(\boldsymbol{\lambda})$, where $\operatorname{diag}(\boldsymbol{\lambda})$ represents the diagonal matrix created by placing the vector $\boldsymbol{\lambda}$ along the diagonal.

First, we see that the derivative of the figure of merit with respect to the shot log-weights is related to the derivative with respect to the shot weights as

$$\frac{\partial \mathcal{F}}{\partial \gamma_T} = \frac{\partial \mathcal{F}}{\partial \Gamma_U}\frac{\partial \Gamma_U}{\partial \gamma_T} = \frac{\partial \mathcal{F}}{\partial \Gamma_U}\left(\Gamma_U\Gamma_T - \delta_{UT}\Gamma_T\right). \tag{2.59}$$

Then recalling Equation 2.28 for the figure of merit, applying the chain rule again yields

$$\frac{\partial \mathcal{F}}{\partial \Gamma_T} = \sqrt{\frac{S'}{N}}\frac{\partial \operatorname{tr}(\Sigma)}{\partial \Gamma_T}\left(\frac{\operatorname{tr}(\Sigma)^2/2 + 3\operatorname{tr}(\Sigma^2)/8}{\operatorname{tr}(\Sigma)^{5/2}}\right) - \sqrt{\frac{S'}{N}}\frac{\partial \operatorname{tr}(\Sigma^2)}{\partial \Gamma_T}\frac{1}{4\operatorname{tr}(\Sigma)^{3/2}}. \tag{2.60}$$

Hence we must determine the derivatives of $\operatorname{tr}(\Sigma)$ and $\operatorname{tr}(\Sigma^2)$ with respect to each of the Γ_T. The chain rule for matrices, as given in [120], yields

$$\frac{\partial \operatorname{tr}(\Sigma)}{\partial \Gamma_T} = \operatorname{tr}\left(\left(\frac{\partial \operatorname{tr}(\Sigma)}{\partial \Omega'}\right)^\intercal \frac{\partial \Omega'}{\partial \Gamma_T}\right), \tag{2.61}$$

$$\frac{\partial \operatorname{tr}(\Sigma^2)}{\partial \Gamma_T} = \operatorname{tr}\left(\left(\frac{\partial \operatorname{tr}(\Sigma^2)}{\partial \Omega'}\right)^\intercal \frac{\partial \Omega'}{\partial \Gamma_T}\right). \tag{2.62}$$

Writing the Uth block of Ω' as an explicit function of the weights and measurement shots S', namely $\Omega'_U = \left(\sum_{V\in\mathcal{T}} \Gamma_V\tau_V/\Gamma_U\right)\Omega^*_U/S'$, we see that

$$\frac{\partial \Omega'_U}{\partial \Gamma_T} = \left(\frac{\tau_T}{\Gamma_U} - \delta_{UT}\frac{\sum_{V\in\mathcal{T}}\Gamma_V\tau_V}{\Gamma_T^2}\right)\frac{\Omega^*_U}{S'}. \tag{2.63}$$

Now it remains only to determine the derivatives of $\operatorname{tr}(\Sigma)$ and $\operatorname{tr}(\Sigma^2)$ with respect to Ω' with the help of the aforementioned algorithmic tool [117, 118], for ordinary, weighted, and generalised least squares estimators.

For ordinary least squares, it is helpful to define $A^*_{\mathrm{OLS}} = \operatorname{diag}(\boldsymbol{\lambda})(A^\intercal A)^{-1}A^\intercal$, so that

we have

$$\Sigma_{\mathrm{OLS}} = A^*_{\mathrm{OLS}}\Omega' A^{*\intercal}_{\mathrm{OLS}} \tag{2.64}$$

$$\frac{\partial \operatorname{tr}(\Sigma_{\mathrm{OLS}})}{\partial \Omega'} = A^{*\intercal}_{\mathrm{OLS}} A^*_{\mathrm{OLS}}, \tag{2.65}$$

$$\frac{\partial \operatorname{tr}(\Sigma^2_{\mathrm{OLS}})}{\partial \Omega'} = 2A^{*\intercal}_{\mathrm{OLS}}\Sigma_{\mathrm{OLS}} A^*_{\mathrm{OLS}}. \tag{2.66}$$

We obtain the same results for generalised least squares, instead defining $A^*_{\mathrm{GLS}} = \operatorname{diag}(\boldsymbol{\lambda})(A^\intercal \Omega'^{-1} A)^{-1} A^\intercal \Omega'^{-1}$ to obtain

$$\Sigma_{\mathrm{GLS}} = A^*_{\mathrm{GLS}}\Omega' A^{*\intercal}_{\mathrm{GLS}} \tag{2.67}$$

$$= \operatorname{diag}(\boldsymbol{\lambda})(A^\intercal \Omega'^{-1} A)^{-1} \operatorname{diag}(\boldsymbol{\lambda}), \tag{2.68}$$

$$\frac{\partial \operatorname{tr}(\Sigma_{\mathrm{GLS}})}{\partial \Omega'} = A^{*\intercal}_{\mathrm{GLS}} A^*_{\mathrm{GLS}}, \tag{2.69}$$

$$\frac{\partial \operatorname{tr}(\Sigma^2_{\mathrm{GLS}})}{\partial \Omega'} = 2A^{*\intercal}_{\mathrm{GLS}}\Sigma_{\mathrm{GLS}} A^*_{\mathrm{GLS}}. \tag{2.70}$$

However, weighted least squares is a little more complicated. We define the diagonal weight matrix with the elementwise Hadamard product as $W = (\Omega' \odot I)^{-1}$, and then both $A^+_{\mathrm{WLS}} = (A^\intercal W A)^{-1} A^\intercal W$ and $A^*_{\mathrm{WLS}} = \operatorname{diag}(\boldsymbol{\lambda}) A^+_{\mathrm{WLS}}$, before finally defining $B_{\mathrm{WLS}} = \Omega' W (A A^+_{\mathrm{WLS}} - I)$. Together, these yield

$$\Sigma_{\mathrm{WLS}} = A^*_{\mathrm{WLS}}\Omega' A^{*\intercal}_{\mathrm{WLS}}, \tag{2.71}$$

$$\frac{\partial \operatorname{tr}(\Sigma_{\mathrm{WLS}})}{\partial \Omega'} = A^{*\intercal}_{\mathrm{WLS}} A^*_{\mathrm{WLS}} \tag{2.72}$$

$$+ 2A^{*\intercal}_{\mathrm{WLS}} A^*_{\mathrm{WLS}} B_{\mathrm{WLS}} \odot I, \tag{2.73}$$

$$\frac{\partial \operatorname{tr}(\Sigma^2_{\mathrm{WLS}})}{\partial \Omega'} = 2A^{*\intercal}_{\mathrm{WLS}}\Sigma_{\mathrm{WLS}} A^*_{\mathrm{WLS}} \tag{2.74}$$

$$+ 4A^{*\intercal}_{\mathrm{WLS}}\Sigma_{\mathrm{WLS}} A^*_{\mathrm{WLS}} B_{\mathrm{WLS}} \odot I. \tag{2.75}$$

Note that the WLS estimator A^+_{WLS} is a left inverse to A, such that $AA^+_{\mathrm{WLS}} - I = [A, A^+_{\mathrm{WLS}}]$ is in fact a commutator. This commutator is zero when A is square and full-rank and therefore admits a right inverse, as in the case of the basic experimental design.

2.E Tuple generation

When generating optimised tuple sets in Section 2.4.3, there are two stages at which we must generate tuples. First, we generate a set $\mathcal{T}_{\mathrm{rep}}$ of tuples which will be repeated some large number of times to amplify and estimate small gate eigenvalues. These resemble the germs of gate set tomography [62]. We later append random tuples to the tuple set, drawing them from some probability distribution over tuples $\mathcal{P}$. In this appendix, we describe how we generate both these repeated and random tuples. We expect it is possible to substantially improve upon the heuristic methods outlined here.

Our tuple generation methods depend on whether the circuit $\mathcal{C}$ is dynamically decoupled. We would like both the repeated and random tuples to reflect this property so as to more closely reflect the original circuit. We find it performant to repeat the layers comprising the repeated tuples such that each repeated tuple implements an involution,

and repeating it twice implements the identity. The repeated tuples for non-dynamically-decoupled circuits are simple and consist of each of the non-empty tuples of the basic tuple set $\mathcal{T}_{\mathcal{I}}$. For dynamically decoupled circuits, single-qubit gate layers are treated in the same manner, whereas layers with multi-qubit gates are interleaved with the dynamical decoupling layer. This ensures that multi-qubit gate layers are not performed in succession, as is the case in the surface code syndrome extraction circuit we aim to characterise. For a concrete example, see the tuples in Table 2.2.

When generating random tuples, which we do by iteratively appending random elements to the tuple, we will often use a discrete power-law distribution with finite support known as the *generalised Zipf distribution*. Up to a normalisation factor, this distribution assigns a weight $1/u^s$ to an outcome u up to a maximum value $u_{\max}$. We will also consider mirror circuits, which implement a series of layers followed by their inverses in reverse order, such that the overall circuit implements the identity. These have recently been used to improve the efficiency of randomised benchmarking [121, 122]. As surface code syndrome extraction circuits consist of layers of X, Hadamard and controlled-X or controlled-Z gates, which are all their own inverses, mirror circuits are generated by mirror tuples, with a simple example of a mirror tuple being $T = (2, 1, 3, 2, 2, 3, 1, 2)$. In the context of ACES, we must append additional layers to mirror circuits in order to obtain full-rank design matrices [74], and accordingly we mirror only the first $L' = \lfloor (L-1)/2 \rfloor$ elements of a length L tuple, leaving one or two non-mirror layers at the end of the tuple for odd or even L, respectively.

To generate a random tuple, we first choose the tuple length L randomly according to a Zipf distribution with $s = 1$ and a maximum length $u_{\max} = 2l$ of twice the length l of the circuit $\mathcal{C}$. We randomly choose whether the tuple will be mirrored or not, and the rule for the number of copies of each index we append to the tuple, each with even probability. Specifically, we append a Zipf-distributed number of copies where we choose either $s = \infty$, which corresponds to appending a single copy, or $s = 2$, which has some probability of appending multiple copies.

For non-dynamically-decoupled circuits, we then simply iteratively append copies of a random index drawn with even probability from the unique layer indices $\mathcal{I}$. It is a little more complicated for dynamically decoupled circuits, as we again want to avoid repeating multi-qubit gate layers. We do this by appending copies of pairs of indices, where the first is drawn before the second to ensure that multi-qubit gate layers are indeed not repeated. When a pair is repeated, the Zipf-distributed repetition number is divided by 2 and rounded up. In the case of mirror circuits, we must also check that a multi-qubit gate layer is not repeated at the point of mirroring.

This heuristic range of rules for generating tuples ensures that we generate a wide variety of random tuples. Despite this, we find that our optimisation algorithm Algorithm 2 generally chooses short random tuples. This is driven at least in part by the fact that the number of repetitions for $\mathcal{T}_{\text{rep}}$ is optimised when augmented by the basic tuple set $\mathcal{T}_{\text{I}}$. We also suspect that the efficient packing of circuit eigenvalue measurements, as specified by Algorithm 1, is a key driver of performance, and this is achieved by mirror tuples, tuples with repeated layers, and shallow tuples. A deeper theoretical understanding of tuple sets should suggest improvements to these methods.

2.F An experimentally-relevant noise model

We would like to examine the performance of our methods on an experimentally-relevant noise model. In this appendix, we construct a distribution over random noise models which resemble and are qualitatively similar to the noise observed in the surface code experiment [11], whose syndrome extraction circuit we depicted in Figure 2.1. We call this distribution log-normal Pauli noise because each non-identity Pauli and measurement error probability is randomly and independently distributed according to some log-normal distribution.

First, we specify the parameters we use for the gate layer, measurement, and reset times. In [11], the overall syndrome extraction circuit is specified to take 921 ns, with measurement taking 500 ns and reset taking 160 ns, leaving 261 ns for the rest of the circuit. Gate layer times are not obviously specified, so we assume they are equal, which is not entirely inconsistent with an earlier experiment [123]. This yields a single-qubit gate layer time $\tau_1 = 29$ ns, a two-qubit gate layer time $\tau_2 = 29$ ns, and a measurement and reset time $\tau_{\text{mr}} = 660$ ns.

For each gate type, including measurements, we tune the mean and variance of the log-normal distribution such that the entanglement infidelity—the sum of a gate's non-identity Pauli error probabilities—has a mean and variance resembling [11]. In particular, we target mean gate infidelities of $r_1 = 0.075\%$ for single-qubit gates, $r_2 = 0.5\%$ for two-qubit gates, and $r_m = 2\%$ for measurements. Note that the single-qubit identity gates used to pad layers are treated the same as other single-qubit gates. Approximating the sum of log-normal distributions as another log-normal distribution with the same mean and variance, we arbitrarily specify that the normal random variable underlying this new distribution has a not dissimilar variance $\sigma_{\text{tot}}^2 = \log(10/9) \approx 0.105$.

A log-normally distributed random variable $X = \exp(\mu_Z + \sigma_Z Z)$, where Z is a normal random variable with mean 0 and variance 1, such that $\mu_Z + \sigma_Z Z$ has mean μ_Z and variance σ_Z^2. Then it is well-known that X has mean $\mu_X = \exp(\mu_Z + \sigma_Z^2/2)$ and variance $\sigma_X^2 = \big(\exp(\sigma_Z^2) - 1\big)\exp(2\mu_Z + \sigma_Z^2)$. We begin by specifying the distribution used to generate measurement error probabilities, as each measurement only has one error probability. Setting $\mu_X = r_m$ and $\sigma_Z^2 = \sigma_{\text{tot}}^2$, rearranging yields $\mu_Z = \log(\mu_X) - \sigma_Z^2/2 = -\log(50) - \log(10/9)/2 \approx -3.965$.

Gates are a little more complicated, as a b-qubit gate has $b' = 4^b - 1$ non-identity Pauli error probabilities. We therefore approximate the sum of these b' log-normal distributions as log-normally distributed with the same mean $\mu_{X;b} = b'\mu_X$ and variance $\sigma_{X;b}^2 = b'\sigma_X^2$. Simple rearrangements yield the mean μ_Z and variance σ_Z^2 of the normal random variable underlying each individual non-identity Pauli error probability as

$$\sigma_Z^2 = \log\big(1 + b'(\exp(\sigma_{Z;b}^2) - 1)\big), \tag{2.76}$$

$$\mu_Z = \log\big(\mu_{X;b}/b'\big) - \sigma_Z^2/2. \tag{2.77}$$

For single-qubit gates where $b = 1$, we set $\sigma_{Z;1}^2 = \sigma_{\text{tot}}^2$ and $\mu_{X;1} = r_1$, yielding $\sigma_Z^2 = \log(4/3) \approx 0.288$ and $\mu_Z = -\log(8000/\sqrt{3}) \approx -8.438$. Similarly for two-qubit gates where $b = 2$, we set $\mu_{X;2} = r_2$ and $\sigma_{Z;2}^2 = \sigma_{\text{tot}}^2$, yielding $\sigma_Z^2 = \log(8/3) \approx 0.981$ and $\mu_Z = -\log(2000\sqrt{6}) \approx -8.497$. As the sum of log-normal distributions is not in fact log-normal, the true variance of the sum will be a little less than our target, with greater deviation for two-qubit gates than for single-qubit gates. Nevertheless, the mean will be consistent with our target.

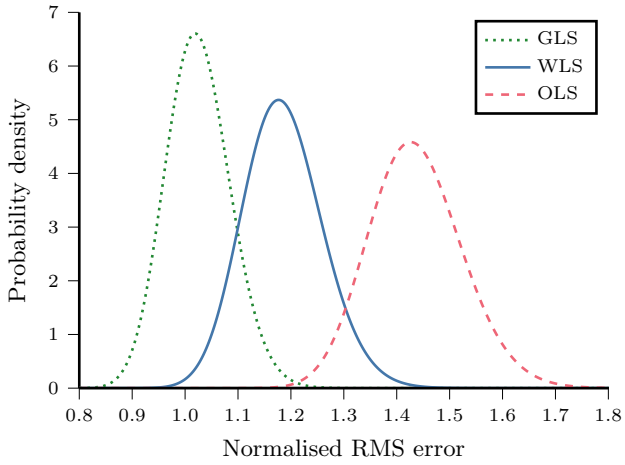

Figure 2.12: Performance comparison of experimental designs optimised for generalised least squares (GLS), weighted least squares (WLS), and ordinary least squares (OLS). Predicted normalised RMS error probability distributions for a fixed-seed random instance of log-normal Pauli noise for the syndrome extraction circuit of a distance-3 surface code.

We can easily compare this to a depolarising Pauli noise model with the same gate and measurement entanglement infidelities r_2, r_1, and r_m. We could think of this as being produced by setting $\sigma_{\text{tot}}^2 = 0$. Be careful to note that under the usual parameterisation, a b-qubit depolarising channel with entanglement infidelity r_b in fact has a depolarising constant $r_b(1 + 1/(4^b - 1))$.

It is easier to calculate quantities for depolarising noise, as they can be calculated directly rather than estimated across random samples of log-normal Pauli noise. Nevertheless, random instances of log-normal Pauli noise are a more interesting and representative target for noise characterisation.

2.G Additional numerical results

We present a number of additional numerical results that were not included in Section 2.5. To further examine these results, refer to the code [88].

The experimental design optimised for depolarising noise on the syndrome extraction circuit of a distance-3 surface code, which we focus on in Section 2.5, is displayed in Table 2.2. Recall that the elements of the tuples index the gate layers appearing in Figure 2.1. We see that while the repeated tuples are very long, the other tuples are very shallow, as suggested by Elfving's theorem. We can also compare our repetitions to the toy example of Section 2.C to check the performance of the repetition number optimisation algorithm. There, the optimal number of Pauli gate repetitions at the same parameter values is 306, compared to 191 here, although only 176 repetitions are required to achieve the asymptotically optimal circuit eigenvalue. These values are similar enough so as to not be a cause for concern, given the different contexts.

Interestingly, most of the measurement budget goes towards measuring the controlled-Z gate layers by themselves, and this is preferred over measuring them alongside single-qubit gate layers. Moreover, no tuple measures multiple controlled-Z gate layers together. This illustrates that performant noise characterisation experimental designs are in fact

Table 2.2: The experimental design with 31 tuples optimised for depolarising noise for the syndrome extraction circuit of a distance-3 surface code shown in Figure 2.1, including the measurement shot weights Γ and tuple set $\mathcal{T}$, as well as the repetition number for the repeated tuples.

Shot weight	Tuple	Repetition number
0.000516	(1)	233
0.000506	(3)	233
0.002424	(5)	191
0.007184	(2, 5, 2, 5)	25
0.007167	(4, 5, 4, 5)	25
0.007138	(6, 5, 6, 5)	25
0.007317	(8, 5, 8, 5)	25
0.158506	(2)	
0.204358	(4)	
0.139123	(6)	
0.074416	(8)	
0.030260	(1, 4)	
0.022677	(1, 6)	
0.029468	(1, 8)	
0.058915	(2, 1)	
0.002187	(2, 5)	
0.045958	(3, 6)	
0.029315	(3, 8)	
0.003016	(5, 2)	
0.002618	(5, 4)	
0.002297	(5, 8)	
0.035342	(6, 1)	
0.032844	(8, 1)	
0.078418	(8, 3)	
0.011389	(2, 5, 3)	
0.003830	(3, 2, 5)	
0.001268	(4, 1, 5)	
0.000428	(4, 5, 5)	
0.000323	(5, 5, 4, 1)	
0.000505	(5, 5, 6, 3)	
0.000466	(5, 6, 3, 5)	

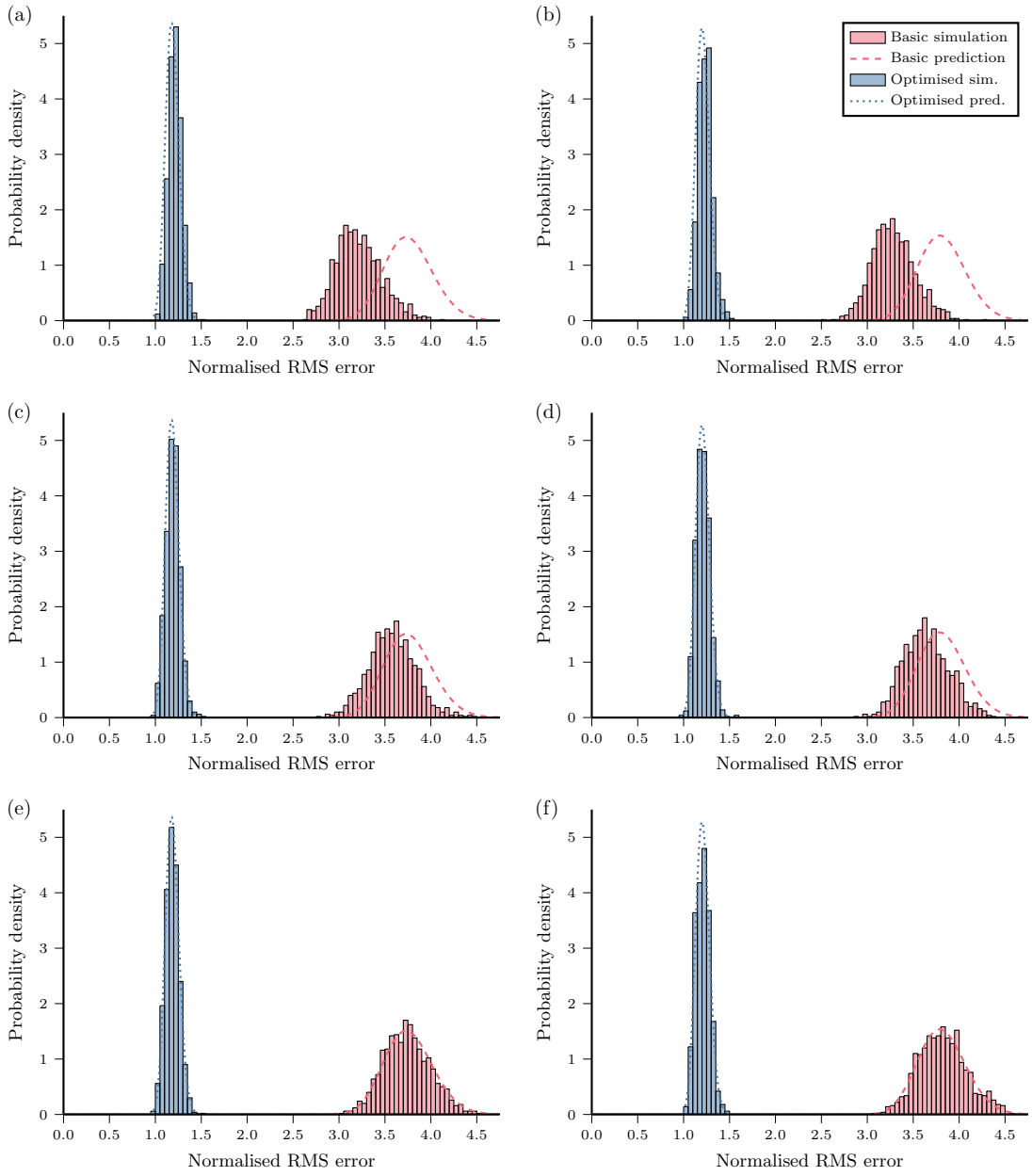

Figure 2.13: Similar results as in Figure 2.3 (reproduced in (e)), but across a range of measurement budgets S and for both a fixed-seed random instance of log-normal Pauli noise and depolarising noise. Specifically, (a, c, e), log-normal Pauli noise; (b, d, f) depolarising noise; (a, b) $S = 10^6$; (c, d) $S = 10^7$; and (e, f) $S = 10^8$.

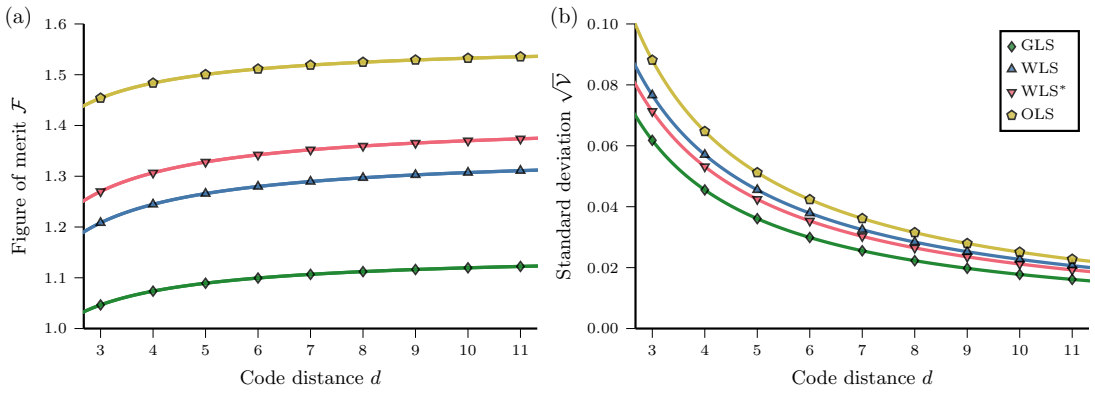

Figure 2.14: Similar results as in Figure 2.4, only for depolarising noise but across a range of experimental designs. The experimental designs are optimised for generalised least squares (GLS), weighted least squares (WLS), and ordinary least squares (OLS), and include one optimised for inappropriate average error rates (WLS*), namely the worst performing design in Figure 2.6. (a) Figure of merit $\mathcal{F}$, the expected normalised RMS error, and (b) the normalised RMS error standard deviation $\sqrt{\mathcal{V}}$ as functions of the surface code distance d, with the data points fit by the same functional forms as in Figure 2.4.

quite simple.

Figure 2.12 compares the predicted performance for experimental designs optimised for the different least squares estimators. As we would expect, GLS performs best, followed by WLS, and then OLS performs worst. Indeed, the expected figures of merit across 400 random instances of log-normal Pauli noise are 1.0346 ± 0.0011, 1.2001 ± 0.0014, and 1.4601 ± 0.0017, respectively. These performance differences are much more substantial than for the designs optimised for inappropriate average error rates.

Then, we perform ACES noise characterisation on the same fixed-seed random instance of log-normal Pauli noise as in Figure 2.3, as well as depolarising noise, comparing the basic and optimised experimental designs. Figure 2.13 show the distribution of the normalised RMS error over 1000 simulated trials of ACES across measurement budgets $S \in \{10^6, 10^7, 10^8\}$ for both noise models. These figures also show the predicted normalised RMS error distributions, which closely align with the simulated data for the largest measurement budget $S = 10^8$ but are overly pessimistic for smaller measurement budgets. As discussed previously, this is because we ensure the parameter estimates are within bounds, which improves performance at small measurement budgets.

Next, we examine how the performance of these optimised experimental designs for depolarising noise varies as a function of the surface code distance d. Specifically, Figure 2.14 shows the figure of merit, the expected normalised RMS error, and the normalised RMS error standard deviation, for the GLS-, WLS-, and OLS-optimised experimental designs and their respective estimators. It also includes the worst performing WLS-optimised experimental design from Figure 2.6. The functional forms again fit the data well, although the GLS figure of merit has much larger relative errors which nevertheless remain under 10^{-3} for each data point. Perhaps this is because the experiment number for the GLS-optimised experimental design is not constant across all distances d, unlike the other designs. Notably, the relative errors for the basic design, which is not

Table 2.3: Condition numbers of the design matrices of optimised experimental designs at a range of surface code distances. The experimental designs are optimised for generalised least squares (GLS), weighted least squares (WLS), and ordinary least squares (OLS), and include the basic design as well as one optimised for inappropriate average error rates (WLS*), namely the worst performing design in Figure 2.6.

Code distance	GLS	WLS	WLS*	OLS	Basic
3	187.98	181.75	380.56	97.57	29.39
4	191.85	185.86	387.92	98.01	30.97
5	193.86	188.13	391.98	98.23	31.70
6	195.17	189.59	394.67	98.35	32.09
7	196.13	190.63	396.61	98.43	32.33
8	196.85	191.39	398.07	98.49	32.48
9	197.40	191.96	399.19	98.53	32.59
10	197.83	192.41	400.07	98.55	32.66
11	198.18	192.77	400.77	98.57	32.71

Table 2.4: Pseudoinverse norms of the design matrices of optimised experimental designs at a range of surface code distances. The experimental designs are optimised for generalised least squares (GLS), weighted least squares (WLS), and ordinary least squares (OLS), and include the basic design as well as one optimised for inappropriate average error rates (WLS*), namely the worst performing design in Figure 2.6.

Code distance	GLS	WLS	WLS*	OLS	Basic
3	0.5994	0.5838	0.5435	0.7061	5.4211
4	0.6002	0.5856	0.5446	0.7064	5.5647
5	0.6013	0.5876	0.5460	0.7066	5.6301
6	0.6026	0.5894	0.5475	0.7068	5.6651
7	0.6039	0.5910	0.5488	0.7069	5.6859
8	0.6050	0.5923	0.5499	0.7070	5.6993
9	0.6060	0.5933	0.5509	0.7071	5.7085
10	0.6068	0.5942	0.5517	0.7072	5.7149
11	0.6075	0.5949	0.5523	0.7072	5.7197

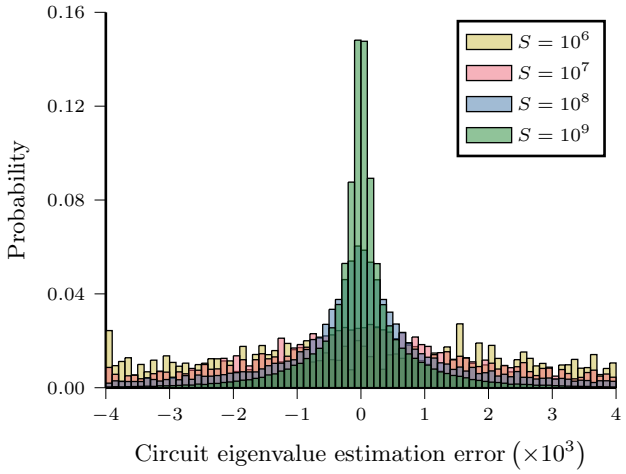

Figure 2.15: Similar results as in Figure 2.7, but characterising depolarising noise rather than a fixed-seed random instance of log-normal Pauli noise. The circuit eigenvalue estimation error distributions are shown for the distance-25 surface code syndrome extraction circuit.

shown here, remain under 10^{-14} for each data point and approach numerical precision.

In Tables 2.3 and 2.4, we show the condition numbers and pseudoinverse norms of the design matrices of these experimental designs at a range of surface code distances d. As with the figure of merit, the condition number and pseudoinverse norm do not substantially increase with the code distance d. Note that at each distance, the condition number for the basic design is the square of the pseudoinverse norm. The pseudoinverse norm was proposed as a candidate figure of merit in [74]. Comparing the optimised and basic experimental designs, we see that the optimisation algorithm decreases the pseudoinverse norm at the cost of increasing the condition number, suggesting that the pseudoinverse norm is a superior figure of merit. However, our figure of merit precisely describes the performance of experimental designs and does not always correlate with the pseudoinverse norm. For example, we see in Table 2.4 that the worst performing design in Figure 2.6 has a better pseudoinverse norm than the best performing design.

Finally, we show the optimised experimental design characterising depolarising noise on the syndrome extraction circuit of a distance-25 surface code. The results are similar to those for a fixed-seed random instance of log-normal Pauli noise. Here, the fits in Figure 2.4 predict a figure of merit, or expected normalised RMS error, of 1.3318, with a normalised RMS error standard deviation of 0.0091. In comparison, the normalised RMS errors were $\{1.3716, 1.3349, 1.3215, 1.3196\}$ across the measurement budgets $S \in \{10^6, 10^7, 10^8, 10^9\}$, which are consistent with the prediction excepting, again, the smallest measurement budget. Figure 2.15 shows the distribution of the circuit eigenvalue estimation error, and Figure 2.16 shows the distribution of the gate estimation error, expressed as the total variation distance (TVD), across the range of gate types appearing in the circuit. We again observe the expected scaling of the median gate estimation errors with the measurement budget S, quantified here in Table 2.5, with improved performance at small measurement budgets as we ensure parameter estimates are within bounds.

Table 2.5: Similar results as in Table 2.1, but instead showing the medians of the gate estimation error distributions in Figure 2.16.

Measurement budget	$-\log_{10}$ median gate estimation error (TVD)			
	Pauli	Hadamard	Measurement	Controlled-Z
10^6	3.847	3.409	3.479	2.719
10^7	4.345	3.782	4.029	3.164
10^8	4.850	4.285	4.541	3.666
10^9	5.336	4.790	5.045	4.168

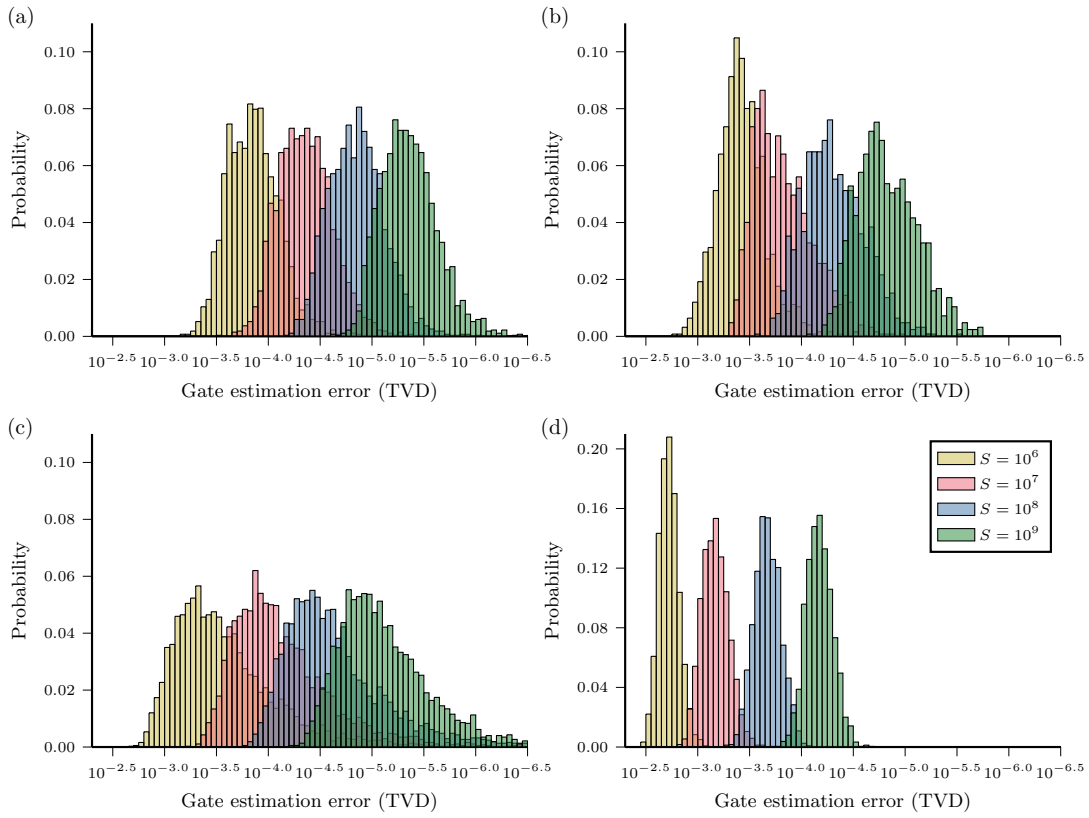

Figure 2.16: Similar results as in Figure 2.8, but characterising depolarising noise rather than a fixed-seed random instance of log-normal Pauli noise. The distributions of the gate estimation error are shown across the range of gate types appearing in the distance-25 surface code syndrome extraction circuit, namely: (a) dynamical decoupling X gates and padded identity gates, or Pauli gates; (b) Hadamard gates; (c) measurements; and (d) controlled-Z gates. Results are shown for the optimised experimental design over a range of measurement budgets S.

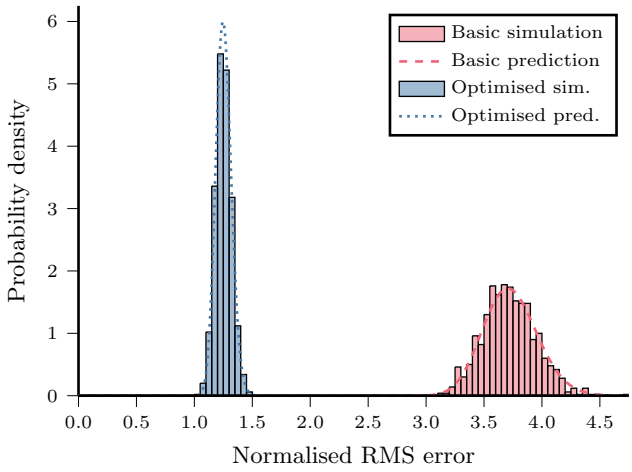

Figure 2.17: Similar results as Figure 2.3, but for the unrotated surface code. Simulated data show the distribution of the normalised RMS error between the estimated and true gate eigenvalues when characterising a fixed-seed random instance of log-normal Pauli noise for the syndrome extraction circuit of a distance-3 unrotated surface code. The data are 1000 trials of characterising a fixed-seed random instance of log-normal Pauli noise with a measurement budget $S = 10^8$ and align with the predicted performance distributions.

2.H ACES for unrotated surface codes

Our numerical results have focused on the syndrome extraction circuit of rotated surface codes. However, we believe our methods apply to the syndrome extraction circuits of arbitrary topological quantum codes. To this end, here we examine the syndrome extraction circuit of unrotated surface codes. Our circuit consists of four layers of controlled-X gates sandwiched between two identical layers of Hadamard gates, such that there are six layers in the circuit and five unique circuit layers. The circuit follows the circuit given in [124] and does not feature dynamical decoupling.

In this appendix, we demonstrate that our methods are also able to characterise the syndrome extraction circuits of unrotated surface codes by reproducing the results in Section 2.5. We obtain remarkably similar results to those for the rotated surface code, despite not modifying our method or parameters. The additional results presented in Section 2.G are also similar for the unrotated surface code, and although we do not reproduce them here, they can be found with the code [88].

We first optimise an experimental design for depolarising noise on the syndrome extraction circuit of a distance-3 unrotated surface code on 25 qubits, using the procedures outlined in Section 2.4. The random tuples added by Algorithm 2 are shallow, with depth at most 4, and the shot weights allocate the majority of the measurement budget to individually measuring the controlled-X gate layers.

Then we perform ACES noise characterisation on a fixed-seed random instance of log-normal Pauli noise, comparing the basic and optimised experimental designs. Figure 2.17 shows the distribution of the normalised RMS error over 1000 simulated trials of ACES with a measurement budget $S = 10^8$ for both designs, as well as the predicted normalised RMS error distributions. The predicted distributions closely align with the simulated

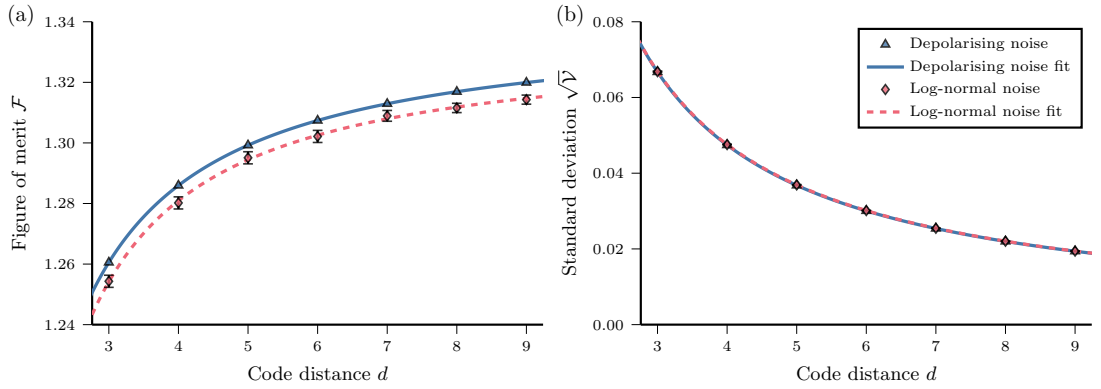

Figure 2.18: Similar results as in Figure 2.4, but for the unrotated surface code. (a) Figure of merit $\mathcal{F}$, the expected normalised RMS error, and (b) the normalised RMS error standard deviation $\sqrt{\mathcal{V}}$, of the optimised experimental design for the syndrome extraction circuit, shown as functions of the unrotated surface code distance d. The data points have no error for depolarising noise, whereas for log-normal Pauli noise, they report the mean estimated by sampling from the distribution over noise models, with error bars indicating two standard deviations. The functional forms fit the data precisely, with relative errors of under 10^{-9} for each depolarising noise data point.

data, validating our ability to predict the performance of ACES noise characterisation experiments, and demonstrating that the optimised design substantially outperforms the basic design. For this random instance of log-normal Pauli noise, the basic design has a figure of merit, or expected normalised RMS error, that is larger than the optimised design by a factor 2.98, implying that the sample efficiency of the optimised design is a factor 8.86 better than the basic design.

Next, we examine how the performance of this optimised experimental design for the syndrome extraction circuit varies as a function of the unrotated surface code distance d. Figure 2.18 depicts the figure of merit $\mathcal{F}$, the expected normalised RMS error, and the normalised RMS error standard deviation $\sqrt{\mathcal{V}}$ as functions of the unrotated surface code distance d. The exact quantities are reported for depolarising noise, whereas for log-normal Pauli noise, we report estimates of the mean across random samples from the distribution over noise models.

Empirically, we find that for depolarising noise, the normalised trace of the gate eigenvalue estimator covariance matrix $\mathrm{tr}\,(\Sigma)/S'$, and of its square $\mathrm{tr}\,(\Sigma^2)/S'^2$, are precisely fit as quadratic functions of d. Model selection using the Akaike information criterion corrected for small samples [109], prefers this quadratic model over other polynomial models. Similarly, the number of gate eigenvalues is exactly described by a quadratic with integer coefficients, $N(d) = 144d^2 - 180d + 54$. Substituting these three quadratics into Equations 2.28 and 2.29 yields functional forms that precisely describe the scaling of $\mathcal{F}$ and $\sqrt{\mathcal{V}}$ with d, which are also depicted in Figure 2.18. For depolarising noise, we fit the normalised traces of the gate eigenvalue estimator covariance matrix, obtaining functional forms for $\mathcal{F}$ and $\sqrt{\mathcal{V}}$ with relative errors of less than 10^{-9} for each data point. By contrast, for log-normal Pauli noise, we directly and simultaneously fit the functional forms for $\mathcal{F}$ and $\sqrt{\mathcal{V}}$. Although the data represent estimates, we nevertheless obtain relative errors of less than 10^{-3} for each data point.

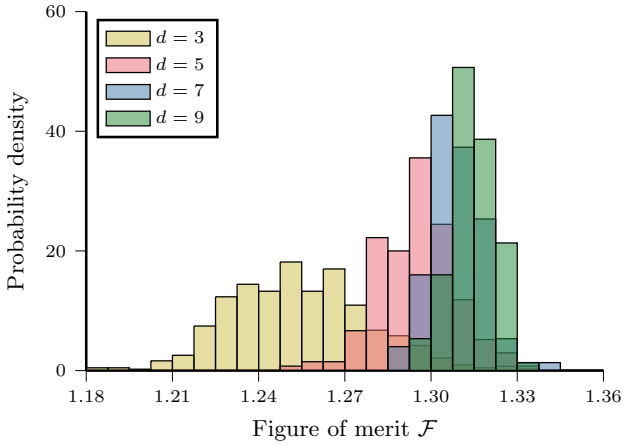

Figure 2.19: Similar results as in Figure 2.5, but for the unrotated surface code. The histograms indicate the distribution of the figure of merit $\mathcal{F}$, the expected normalised RMS error, of the optimised experimental design for the syndrome extraction circuit, across random instances of log-normal Pauli noise. Results are shown across a range of unrotated surface code distances d.

Importantly, the functional forms derived by substituting the quadratics into Equations 2.28 and 2.29 entail that in the limit of large d, the figure of merit $\mathcal{F}$, or expected normalised RMS error, approaches a constant value. Also, the optimised experimental design requires 219 experiments, before Pauli frame randomisation, to estimate all of the circuit eigenvalues at all tested code distances d, which range from $d = 3$ to $d = 17$. This demonstrates that ACES is capable of estimating noise in surface code syndrome extraction circuits to a precision that is asymptotically independent of the number of qubits n in the code, using a number of experiments that is also asymptotically independent of n.

Notably, the optimised experimental design performs better on average when characterising log-normal Pauli noise, despite being optimised for depolarising noise. Moreover, Figure 2.19 shows histograms of the figure of merit $\mathcal{F}$ across random instances of log-normal Pauli noise for the syndrome extraction circuit at a range of unrotated surface code distances d. It shows that random samples of log-normal Pauli noise have increasingly similar figures of merit as d increases. Together, these suggest that the performance of the optimised experimental design is not substantially harmed by small changes in the noise model that preserve average error rates. The effects of randomness in the noise model on the performance of ACES also appear to average out with increasing system size.

We can also examine experimental designs optimised at different error rates to determine the robustness of their performance. Specifically, we optimise for half the error rate, the error rate, and double the error rate, on single-qubit gates, two-qubit gates, and measurements, yielding 27 different experimental designs. Figure 2.20 shows the expected number of samples required to achieve a fixed estimation accuracy, evaluated over 400 random instances of log-normal Pauli noise for the syndrome extraction circuit of a distance-3 unrotated surface code. These values are expressed relative to the original experimental design optimised for the appropriate average error rates, which performs best.

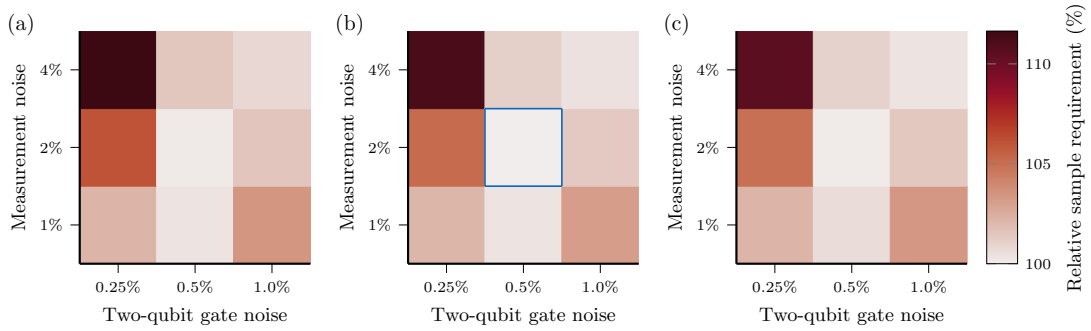

Figure 2.20: Similar results as in Figure 2.6, but for the unrotated surface code. The experimental designs were optimised for depolarising noise with a range of error rates, with single-qubit gate error rates (a) $r_1 = 0.0375\%$, (b) $r_1 = 0.075\%$, and (c) $r_1 = 0.15\%$, and two-qubit gate and measurement error rates indicated in the heatmaps. Experimental designs were evaluated for the syndrome extraction circuit of a distance-3 unrotated surface code against 400 random instances of log-normal Pauli noise, whose average error rates are indicated by the blue square. Heatmap colour indicates the expected number of samples required to achieve a fixed estimation accuracy, expressed relative to the best-performing design optimised at the correct average error rates. Only three experimental designs require over 10% more samples to achieve the same accuracy as the best-performing design.

Only 5 of the 27 experimental designs require over 5% more samples than the most performant experimental design to achieve the same expected accuracy, with the worst requiring $11.64 \pm 0.10\%$. This corresponds to having a mean figure of merit of 1.3236 ± 0.0014, whereas the mean figure of merit of the most performant design is 1.2527 ± 0.0012. The expected relative number of samples required to achieve the same accuracy is the square of this ratio, and is estimated precisely due to substantial covariance in the figure of merit between designs for each instance of log-normal Pauli noise. Overall, this demonstrates that the performance of optimised experimental designs is robust to being optimised for inappropriate average error rates.

Finally, we can use our optimised experimental design to characterise a fixed-seed random instance of log-normal Pauli noise on the syndrome extraction circuit of a distance $d = 17$ unrotated surface code with $n = 1089$ qubits. The optimised experimental design features 25 tuples which together estimate $195,723$ circuit eigenvalues over 219 experiments, before Pauli frame randomisation, in order to estimate the $38,610$ gate eigenvalues. Optimising the experimental design at $d = 3$, generating it at $d = 17$, and then simulating ACES noise characterisation experiments for measurement budgets $S \in \{10^6, 10^7, 10^8, 10^9\}$ together took under 3 hours, with roughly half of that time being dedicated to stabiliser circuit simulations with Stim. The fits in Figure 2.18 predict an average figure of merit 1.3249 and a standard deviation 0.010, across random instances of log-normal Pauli noise. By comparison, for the simulated noise characterisation of a specific random instance of log-normal Pauli noise, the normalised RMS errors were $\{1.3135, 1.3207, 1.3237, 1.3181\}$ across measurement budgets. These results are consistent with our performance predictions.

The noise estimation procedure begins by estimating the circuit eigenvalues from raw measurement data. Figure 2.21 depicts histograms of the circuit eigenvalue estimation

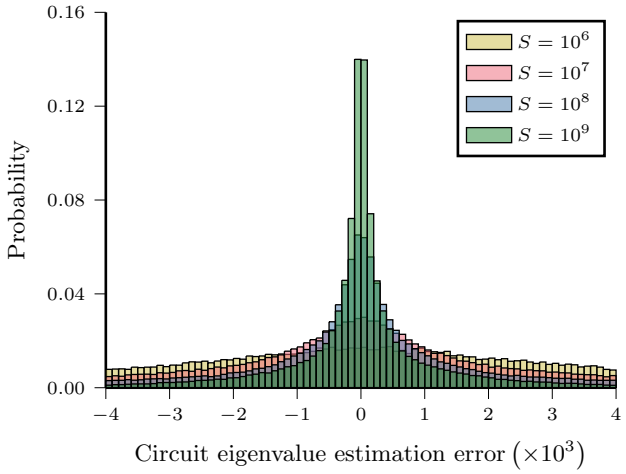

Figure 2.21: Similar results as in Figure 2.7, but for the syndrome extraction circuit of a distance-17 unrotated surface code. The histograms indicate distributions of the circuit eigenvalue estimation error (the difference between the estimated and true circuit eigenvalues). The data are for a fixed-seed random instance of log-normal Pauli noise over a range of measurement budgets S.

error, the difference between the estimated and true circuit eigenvalues, across the range of measurement budgets. As the measurement budget increases, the distributions of the circuit eigenvalue estimation error narrow, straightforwardly improving the accuracy of the circuit eigenvalue estimates.

Ultimately, the protocol estimates the Pauli error probability distribution of all the gates and measurements appearing in the circuit. We measure the gate estimation error with the total variation distance (TVD) between the estimated and true Pauli error probability distribution, a principled measure for probability distributions and half the diamond norm between Pauli channels [94]. Figure 2.22 depicts histograms of the gate estimation error across measurement budgets and the different types of gate appearing in the circuit, namely: padded identity gates, which are Pauli gates; Hadamard gates; measurements; and controlled-X gates. The Pauli gates tend to be estimated to a higher precision than the Hadamard gates, despite both being single qubit gates. By contrast, the two-qubit controlled-X gates are least accurately estimated, accounted for in part by their Pauli error probability distribution being over 16 errors, compared to 4 for the single-qubit gates and just 2 for measurements.

The gate estimation errors improve roughly by a constant factor of $\sqrt{10}$ for each factor of 10 increase in the measurement budget S, demonstrating the expected $1/\sqrt{S}$ sample efficiency. This is clear visually and quantified in Table 2.6, which shows the median gate estimation error across gate types and measurement budgets. The Hadamard and controlled-X gate estimates outperform the trend, particularly for the smallest measurement budget $S = 10^6$. This is because large estimation errors for small values of S are improved by ensuring parameter estimates are within bounds. Specifically, gate eigenvalues greater than 1 are set to 1, and estimated probability distributions are projected into the simplex. For the data in Table 2.6, at $S = 10^6$, 460 gate eigenvalues are set to 1, while at $S = 10^7$, only 17 gate eigenvalues are set to 1.

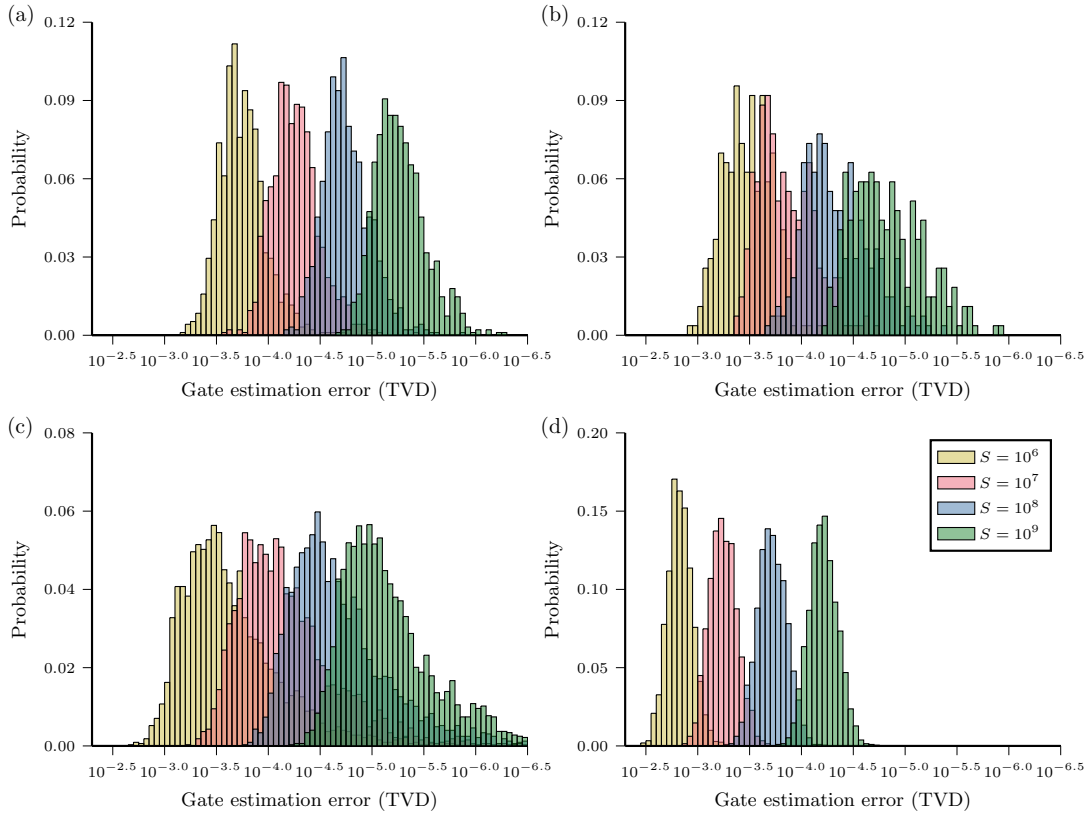

Figure 2.22: Similar results as in Figure 2.8, but for the syndrome extraction circuit of a distance-17 unrotated surface code. The histograms indicate distributions of the gate estimation error (the total variation distance (TVD) between the estimated and true error probability distribution for the gate). The data are for a fixed-seed random instance of log-normal Pauli noise over a range of measurement budgets S. The estimation error distributions are shown across the range of gate types appearing in the circuit: (a) dynamical decoupling X gates and padded identity gates, or Pauli gates; (b) Hadamard gates; (c) measurements; and (d) controlled-Z gates.

Table 2.6: Similar results as in Table 2.1, but instead showing the medians of the gate estimation error distributions in Figure 2.22.

Measurement budget	$-\log_{10}$ median gate estimation error (TVD)			
	Pauli	Hadamard	Measurement	Controlled-X
10^6	3.728	3.466	3.538	2.826
10^7	4.245	3.808	4.083	3.238
10^8	4.725	4.290	4.585	3.712
10^9	5.241	4.776	5.071	4.207

Overall, we see that the results for the rotated and unrotated surface code syndrome extraction circuits are very similar, supporting the utility of our methods for characterising noise in the syndrome extraction circuits of general topological quantum codes, as well as other fault-tolerant gadgets.

Chapter 3

Improving error suppression with noise-aware decoding

> What is true is already so. Owning up to it doesn't make it worse. Not being open about it doesn't make it go away. And because it's true, it is what is there to be interacted with.
>
> — Eugene Gendlin, *Focusing*

Abstract

We demonstrate that the performance of quantum error correction can be improved with noise-aware decoders that are calibrated to the likelihood of physical error configurations in a device. We show that noise-aware decoding increases the error suppression factor of the surface code, yielding reductions in the logical error rate that increase exponentially with the code distance. Our calibration protocol involves circuit-level Pauli noise characterisation experiments with averaged circuit eigenvalue sampling. This enables decoder calibration at the scales required for fault-tolerant quantum computation and near-optimal decoding when compared to the true noise model. Our results indicate that these noise characterisation experiments could be performed and processed in seconds for superconducting quantum computers. This establishes the practicality and utility of noise-aware decoding for quantum error correction at scale.

3.1 Introduction

The need for real-time decoding of error syndrome data in fault-tolerant quantum computers will limit decoders and hence the performance of quantum error correction. Decoders attempt to infer underlying physical errors from error syndrome data to determine a correction operation that preserves the logical information. They must do so as quickly as the error syndrome data accumulate, or else the backlog grows and slows computation exponentially [2]. Consequently, fault-tolerant quantum computation will require fast decoders, such as [45, 83, 125, 126], capable of real-time operation [17, 127], even if they are less accurate than alternatives [19, 43].

Mechanisms for improving decoder performance without increasing computational cost are therefore an important factor in reducing fault tolerance overheads. One such mechanism is noise-aware decoding, which improves performance by calibrating decoders with estimates of the noise in the quantum device. Calibrating the decoder prior on the likelihood of error configurations allows more accurate inference of physical errors from error syndrome data [77, 81, 83, 84, 128]. Decoder priors can also be learned online from error syndrome data [129–132], or directly optimised to minimise the logical error rate of the code [133].

Averaged circuit eigenvalue sampling (ACES) [74] is a scalable method for characterising the Pauli noise associated with the operation of all gates in the syndrome extraction circuits of large quantum error correcting codes [134]. Recently, there has been rapid experimental progress in quantum error correction [11, 12, 14–18, 54, 55], generating an immediate need for practical and scalable methods for calibrating the decoder prior. We use ACES estimates of circuit-level Pauli noise to calibrate a fast correlated matching decoder [45]. These estimates can also be used to generate simulated data for training more accurate decoders, such as in [19], to verify appropriate device calibration, and to inform the tailoring of codes and decoders to noise biases [48, 79, 80, 135, 136].

In this letter, we establish the practicality of decoder calibration by demonstrating that ACES enables near-optimal noise-aware decoding when compared to the true noise model. We show through circuit-level numerical simulations that noise-aware decoding increases the error suppression factor of the surface code [5, 49–51], yielding reductions in the logical error rate that increase exponentially with the code distance. We find that decoder calibration substantially reduces logical error rates and qubit overheads at the scales necessary for fault-tolerant quantum computation. Our results indicate that the requisite noise characterisation experiments could be performed and processed in seconds for superconducting quantum computers. This establishes the practicality and utility of noise-aware decoding for quantum error correction at scale.

3.2 Introducing ACES

We now introduce the fundamental ideas underlying averaged circuit eigenvalue sampling (ACES) [74]. ACES enables scalable characterisation of the Pauli noise associated with implementing the syndrome extraction circuits of topological quantum codes [134]. For a detailed treatment of the ACES protocol used here, refer to [134].

Quantum devices typically implement syndrome extraction circuits by performing a sequence of layers of gates. A *gate* is a primitive operation on the quantum device, typically acting on one or two qubits, and a *layer* is a set of gates that are implemented simultaneously and act on disjoint sets of qubits. Syndrome extraction circuits are typically implemented with gates drawn from the Clifford group, with some examples being the controlled-X, Hadamard, and phase gates [32]. A defining property of the Clifford group is that its elements map Pauli operators to Pauli operators. Clifford circuits, also known as stabiliser circuits, can be efficiently simulated [31, 32]. We model noise with Pauli channels

$$\mathcal{E}(\rho) = \sum_{\boldsymbol{a} \in \mathbf{P}^n} p_{\boldsymbol{a}} P_{\boldsymbol{a}} \rho P_{\boldsymbol{a}}, \tag{3.1}$$

which act on quantum states by applying a probabilistic mixture of Pauli operators characterised by the *Pauli error probabilities* $p_{\boldsymbol{a}}$. We represent quantum noise in this way

because the technique of Pauli frame randomisation tailors arbitrary noise channels into *Pauli channels* [36, 90].

ACES characterises Pauli noise associated with Pauli frame randomised Clifford circuits. It focuses on a *circuit-level Pauli noise model* by estimating the Pauli error probabilities of the noise channel associated with each gate in the circuit operated in the specific context of its layer. ACES does this by measuring Pauli observables for circuits arranged from the same layers as the Clifford circuit of interest. The *circuit eigenvalue* Λ_μ is precisely the expectation value of a Pauli observable for a rearranged circuit, where μ indexes both the Pauli observable and circuit rearrangement. Classically simulating the action of the circuit on the relevant input states allows us to decompose the circuit eigenvalue into a product of *gate eigenvalues* λ_ν, where ν indexes both the gate and the state of the observable before it is acted upon by the gate. These are, in fact, the eigenvalues of the Pauli noise channel associated with the gate [35]. These gate eigenvalues are related to the Pauli error probabilities by an invertible linear transform called the Walsh-Hadamard transform.

Performing stabiliser circuit simulations allows us to turn this into a linear regression problem by creating a *design matrix*. The design matrix entries $A_{\mu\nu}$ are the powers of the gate eigenvalue λ_ν in the decomposition of the circuit eigenvalue Λ_μ. Taking the logarithm, we obtain the linear system of equations

$$\log \Lambda_\mu = \sum_\nu A_{\mu\nu} \log \lambda_\nu. \tag{3.2}$$

ACES thus reduces the problem of Pauli noise characterisation to estimating a set of circuit eigenvalues such that this design matrix has full rank. Then we perform linear least squares to obtain estimates of the gate eigenvalues and hence the Pauli error probabilities of each of the gates in the circuit of interest. We can optimise the design of ACES noise characterisation experiments to improve the sample efficiency of their noise estimates, as described in [134]. Section 3.A outlines a number of improvements to the ACES protocol of [134].

3.3 Quantum codes and decoders

Quantum error correcting codes typically work by repeatedly performing parity check measurements [2], which may or may not mutually commute in the respective cases of stabiliser codes [31] and subsystem codes [53]. These parity check measurements are implemented by repeatedly performing a *syndrome extraction circuit*. A *detector* of such a circuit is a parity of measurement outcomes—a product of parity check measurements—that is deterministic in the absence of noise. A detector is said to be flipped if its value differs from the deterministic outcome.

We focus on surface codes [5, 49–51], a well-studied family of stabiliser codes that encode a single logical qubit in a $d \times d$ square array of physical qubits called data qubits. The parameter d is the distance of the code. A distance $d = 5$ surface code is depicted in Figure 3.1 alongside the syndrome extraction circuit we consider in this letter. Also shown in Figure 3.1 are the $d^2 - 1$ measure qubits used by this circuit to measure the Pauli Z and X stabilisers of the code. We operate the surface code in a memory experiment by repeatedly performing its syndrome extraction circuit. The detectors in a memory experiment are the parities of stabiliser measurement outcomes across consecutive rounds of syndrome extraction.

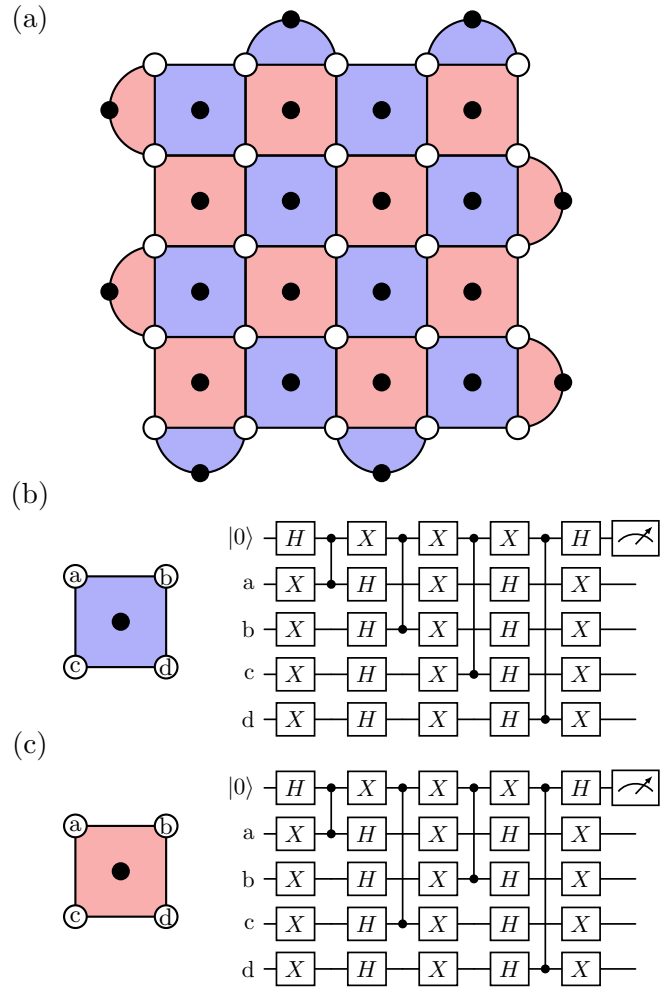

Figure 3.1: (a) Diagram of a distance $d = 5$ rotated surface code. Data qubits are shown as open circles, and measure qubits are shown as filled circles placed in their corresponding stabiliser plaquettes. Syndrome extraction circuit components are shown for (b) blue and (c) red stabiliser plaquettes, respectively. These circuits implement the XZZX variant of the surface code [47, 48], following recent experiments [11, 16].

Stabiliser simulation of the syndrome extraction circuit with a circuit-level Pauli noise model, which ACES can estimate, allows us to efficiently calculate the *decoder prior*. The decoder prior describes the probability of each possible combination of detector flips caused by Pauli error mechanisms in the circuit. Then the *syndrome* of an error is the set of detectors flipped by that error.

A decoder infers a correction operation from this syndrome data that aims to preserve the logical information in the code. Fast decoders avoid the decoding backlog problem [2], enabling quantum error correction at scale. We focus on fast minimum-weight perfect matching (MWPM) decoders [45, 50, 125, 126], which can operate in constant parallel time [125, 126]. MWPM decoders infer the most probable error consistent with the syndrome. This probability is defined by the decoder prior, and so accurate calibration of this prior is important—not only for MWPM decoders but indeed for all decoders.

A memory experiment examines how well a quantum error correcting code preserves

logical information, such as Pauli X or Z logical observables. We will examine surface code memory experiments characterised by the code distance d and rounds of syndrome extraction r, following the circuits in [11, 16]. When the noise associated with implementing the syndrome extraction circuit is sufficiently low, the code is said to be operating below its threshold [51]. Below threshold, the logical error rate per round of syndrome extraction ε is well-described by the approximate scaling relation $\varepsilon \propto \Lambda^{-d/2}$. The *error suppression factor* Λ characterises the error suppression associated with increasing the code distance by 2 and depends on the decoder. We are interested in comparing the error suppression factors when the decoder is calibrated with different priors, as error suppression factor improvements correspond to exponential reductions in the logical error rate.

3.4 Numerical results

We now present numerical results demonstrating the utility of calibrating the decoder prior and the practicality of doing so with ACES. These results were produced on a 2021 M1 Max Macbook Pro with 32 GB of RAM. We use Stim for stabiliser circuit simulations [56], PyMatching for a fast correlated matching decoder [45, 57], and QuantumACES to design and simulate ACES noise characterisation experiments [88].

We test our methods on a distribution over noise models called log-normal Pauli noise, where the Pauli error probabilities of each gate are independently log-normally distributed, following [134]. We use a physically relevant noise parameter regime resembling a recent experiment that operated the surface code below threshold [16]. In particular, we choose average single-qubit, two-qubit, measurement and measurement idle, and reset error rates of $r_1 = 0.05\%$, $r_2 = 0.4\%$, $r_m = 0.8\%$, and $r_r = 0.2\%$, and standard deviations $\sigma_1 = \sigma_2 = 1/2$ and $\sigma_m = \sigma_r = 1/4$ for the underlying normal distributions. For data reproducibility, we fix the seed to be 0 when drawing only a single random instance from this distribution over noise models. We will also consider tuned depolarising noise, where component error rates are set to be their average values for log-normal Pauli noise, and optimise our ACES experimental design according to these parameters for a distance-3 surface code syndrome extraction circuit.

First, we examine the performance of surface code memory experiments, averaged over random instances of log-normal Pauli noise, across a range of decoder priors and code distances. These results were obtained by fitting the exponentially decaying logical error rate, estimated over 10^5 shots, over rounds $r \in \{3, 5, 9, 17, 33\}$ in both X and Z memory experiments. We average over both the memory experiment type and $\{1500, 300, 100, 80, 60, 50\}$ random instances of log-normal Pauli noise for distances $\{3, 5, 7, 9, 11, 13\}$, respectively. In each instance, decoding was performed with priors derived from the true noise model, tuned depolarising noise, and ACES noise estimates with 10^6 and 10^7 shots. Fitting the logical error per round as an exponential in the code distance yields average error suppression factor estimates $\Lambda_{\text{true}} = 1.7360 \pm 0.0025$ for the true noise model, $\Lambda_{\text{dep}} = 1.6967 \pm 0.0025$ for tuned depolarising noise, and $\Lambda_{\text{ACES:}10^6} = 1.7347 \pm 0.0025$ and $\Lambda_{\text{ACES:}10^7} = 1.7358 \pm 0.0025$ for ACES noise estimates with 10^6 and 10^7 shots, respectively. We find that decoding with ACES noise estimates performs similarly to decoding with the true noise model, even with only 10^6 shots, whereas decoding with depolarising noise performs significantly worse.

Importantly, noise-aware decoding increases the error suppression factor, leading to

reductions in the logical error rate that increase exponentially with the code distance. Intuitively, as the code distance increases, tolerable error configurations become larger and more complicated, increasing the freedom in decoding. Hence we would expect the benefit of calibrating the decoder prior to increase with scale as we see here. This motivates a focus on highly scalable methods for calibrating decoder priors such as ACES.

Figure 3.2 displays the relative performance of different decoder priors averaged over random instances of log-normal Pauli noise, expressing the logical error per round for different decoder priors relative to tuned depolarising noise. The covariance of the logical error rates between decoder priors across each instance of log-normal Pauli noise allows us to precisely estimate error suppression factors relative to the true noise model prior. Then the data are normalised by the fit for tuned depolarising noise to highlight the performance gains of noise-aware decoding. These results show that decoding with ACES noise estimates allows near-optimal decoding as compared to the true noise model, whereas decoding with tuned depolarising noise noticeably decreases performance.

Next, we test noise-aware decoding at scale by examining surface code memory experiments at the larger distance $d = 25$. To begin, we simulate a memory experiment for a single random instance of log-normal Pauli noise with $r = 25$ rounds, dividing 10^7 shots evenly between X and Z memory experiments. The decoder confusion matrix in Table 3.1 compares the raw error counts when decoding with the true noise model, tuned depolarising noise, and weighted least squares ACES noise estimates with 10^6 and 10^7 shots. Each comparison between decoder priors shows it is more common for decoding

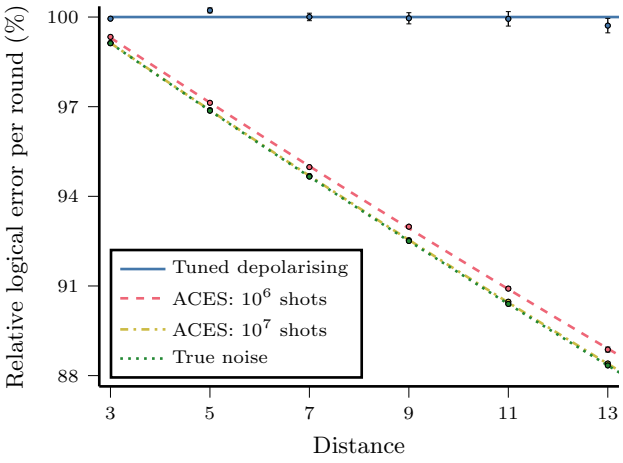

Figure 3.2: Average logical error rate per round of surface code syndrome extraction across priors for the MWPM decoder expressed relative to tuned depolarising noise as a function of the code distance. Results are averaged over random instances of log-normal Pauli noise and X and Z memory experiments and the MWPM decoder prior is derived from the true noise model, tuned depolarising noise, and ACES noise estimates obtained with 10^6 and 10^7 shots. The data points are fit to an exponential in the code distance, error bars indicate one standard deviation, and all values are expressed with respect to the fit for tuned depolarising noise. The performance gain from decoding with ACES noise estimates rather than tuned depolarising noise grows linearly with the code distance, and ACES noise estimates with 10^7 shots obtain almost all of the performance gains associated with decoding with the true noise model.

Table 3.1: Decoder performance for distance-25 surface code memory X and Z experiments with 25 rounds under a random instance of log-normal Pauli noise, dividing 10^7 shots evenly between the memory types. The MWPM decoder prior is derived from the true noise model, ACES noise estimates with 10^7 and 10^6 shots, and tuned depolarising noise, ordered by the accuracy and decoding performance of the noise model. Diagonal elements count decoding failures for each prior. Off-diagonal elements count the number of shots where the decoder for the row succeeded and the decoder for the column failed. Decoding with ACES noise estimates achieves similar logical error rates to decoding with the true noise model.

Fail. \ Fail. / Succ.	True	ACES:10^7	ACES:10^6	Depolarising
True	**5507**	227	619	3005
ACES:10^7	195	**5539**	564	2997
ACES:10^6	495	472	**5631**	2994
Depolarising	1314	1338	1427	**7198**

with the more accurate noise model to succeed when the less accurate noise model fails, though the converse does occur.

We also fit the logical error per round estimated over rounds $r \in \{3, 5, 9, 17, 33\}$ with 10^6 shots for both X and Z memory experiments, averaging over the memory experiment type. We obtain logical error per round estimates $\varepsilon_{25,\text{true}} = (2.39 \pm 0.05) \times 10^{-5}$ for the true noise model, $\varepsilon_{25,\text{dep}} = (3.13 \pm 0.06) \times 10^{-5}$ for tuned depolarising noise, and $\varepsilon_{25,\text{ACES:}10^6} = (2.42 \pm 0.05) \times 10^{-5}$ and $\varepsilon_{25,\text{ACES:}10^7} = (2.40 \pm 0.05) \times 10^{-5}$ for ACES noise estimates with 10^6 and 10^7 shots, respectively. Now we can compare relative logical errors per round for this single random instance of log-normal Pauli noise to the extrapolated trends of Figure 3.2 which instead describe the average over random instances of log-normal Pauli noise. We have $\varepsilon_{25,\text{dep}}/\varepsilon_{25,\text{true}} = 1.31 \pm 0.04$ compared to the predicted average value 1.300 ± 0.004, $\varepsilon_{25,\text{ACES:}10^6}/\varepsilon_{25,\text{true}} = 1.01 \pm 0.03$ compared to 1.012 ± 0.001, and $\varepsilon_{25,\text{ACES:}10^7}/\varepsilon_{25,\text{true}} = 1.00 \pm 0.03$ compared to 1.001 ± 0.001. Hence the improvement from noise-aware decoding for this single random instance of log-normal Pauli noise at a large scale is highly consistent with the extrapolated trends for the average over instances of log-normal Pauli noise at small scales. We attribute this to a self-averaging effect.

Given this confidence in the extrapolated trends of Figure 3.2, we can now examine the logical error rate and qubit reductions achievable at the scales necessary for quantum computation. For example, at distance $d = 63$, decoding with tuned depolarising noise is predicted to obtain a logical error rate of approximately $\varepsilon = 10^{-9}$. We find $\varepsilon_{63,\text{true}} = (5.169 \pm 0.009) \times 10^{-10}$ for the true noise model, $\varepsilon_{63,\text{dep}} = (10.037 \pm 0.019) \times 10^{-10}$ for tuned depolarising noise, and $\varepsilon_{63,\text{ACES:}10^6} = (5.296 \pm 0.009) \times 10^{-10}$ and $\varepsilon_{63,\text{ACES:}10^7} = (5.191 \pm 0.009) \times 10^{-10}$ for ACES noise estimates with 10^6 and 10^7 shots, respectively. At this distance, noise-aware decoding is predicted to roughly halve the logical error rate. Indeed, at distance $d = 61$, we predict $\varepsilon_{61,\text{ACES:}10^6} = (9.187 \pm 0.015) \times 10^{-10}$. This represents a simultaneous reduction in the logical error rate and qubit overhead, from $n = 7937$ at $d = 63$ to $n = 7441$ at $d = 61$, a relative qubit reduction of over 6% and an

absolute reduction of 496 qubits.

3.5 Conclusions

In this letter, we have demonstrated that ACES is practically capable of calibrating decoder priors. We find that noise-aware decoding increases the error suppression factor of the surface code, exponentially reducing logical error rates compared to decoding with a tuned depolarising noise model. Calibration with ACES noise estimates enables near-optimal decoding compared to calibration with the true noise model. Our findings indicate that noise-aware decoding can substantially reduce logical error rates and qubit overheads at the scales necessary for quantum computation. As the tuned depolarising noise model we use as a comparison is itself informed by accurate estimates of the average error rates, gains may be larger in practice.

Noise estimates obtained from ACES with 10^6 shots appear to be sufficiently precise for the purpose of calibrating decoder priors. In the surface code experiment referenced for our noise model parameter regime [16], gate, measurement, and reset times indicate that ACES experiments with the design featured in this letter could collect 10^6 shots in about two seconds with an appropriate hardware and software control stack. In our laptop-based numerical simulations of ACES for a distance-25 surface code syndrome extraction circuit, classical processing of the data with ACES can be performed in under four seconds. This processing time could be substantially reduced by improved hardware and optimised software. Therefore, we expect that decoder priors could be calibrated with ACES experiments performed and processed in seconds for superconducting quantum computers.

A natural next step would be implementing these methods in a memory experiment performed on a quantum device. However, as we find that the key benefit of noise-aware decoding is in improving the error suppression factor of the code, performance improvements may be difficult to realise in small codes. It would be instructive to more thoroughly examine the calibration of decoder priors with ACES across different error rates and decoders. It would also be interesting to compare the performance and practicality of calibration methods based on noise characterisation, such as ACES, with those that directly attempt to minimise the logical error rate, for example with reinforcement learning [133]. Moreover, ACES noise estimates could be used to inform simulations that generate synthetic data that could then be included in the training data for neural network decoders, such as [19], which may improve performance.

An important advantage of methods such as ACES, which use estimates of circuit-level noise to calibrate decoder priors, is that they can also diagnose device performance and identify poorly performing gates and measurements. Decoder priors calibrated in this way could then be updated online using syndrome data [129–132], accounting for device drift. We hope this work paves the way for a sophisticated yet practical stack of methods for rapid calibration of decoder priors at scale in fault-tolerant quantum computing architectures.

Acknowledgements

We thank Nicholas Fazio for discussions. This work was supported by the Australian Research Council Centre of Excellence for Engineered Quantum Systems (CE170100009)

and the U.S. Army Research Office (W911NF-23-S-0004).

Data availability

The code supporting the findings of this letter is openly available in QuantumACES [88].

3.A Relative precision ACES

In this appendix, we briefly outline a number of modifications we make to the ACES protocol described in [134], enabling relative precision noise estimation. We build upon the notation of [134] and do not separately introduce it here. Our modifications follow Pauli noise characterisation theory [96, 115], also outlined in cycle error reconstruction [76].

The key idea is to estimate gate eigenvalues corresponding to the *orbits* of a gate, sets of Paulis mapped to each other by the action of the gate, rather than for all of the Paulis. Given some Clifford gate $\mathcal{G}$, let the $\mathcal{G}$-*orbit* of a Pauli $P_{\boldsymbol{a}}$ be

$$P_{\boldsymbol{a}}^{\mathcal{G}} = \{\mathcal{G}^j(P_{\boldsymbol{a}}) \colon j \in \mathbb{N}\}, \tag{3.3}$$

where we neglect the sign of the Paulis. Then the $\mathcal{G}$-*orbits* are simply the $\mathcal{G}$-orbits of each of the Paulis supported on $\mathcal{G}$, which we write as

$$\text{Orbit}\,(\mathcal{G}) = \{P_{\boldsymbol{a}}^{\mathcal{G}} \colon \boldsymbol{a} \in \text{Pauli}\,(\mathcal{G})\}. \tag{3.4}$$

Note that $|\text{Orbit}\,(\mathcal{G})| \leq |\text{Pauli}\,(\mathcal{G})|$, which is strict if and only if there exist Paulis on which the gate acts non-trivially up to sign, so that equality is achieved by Pauli gates.

Now consider the circuit eigenvalue corresponding to preparing the Pauli $P_{\boldsymbol{a}}$ and then repeating the gate $\mathcal{G}$ t times, which we will write as $\Lambda_{\mathcal{G}^{(t)},\boldsymbol{a}}$. Now divide t by the size of the $\mathcal{G}$-orbit of $P_{\boldsymbol{a}}$ to obtain integers u, v such that $t = |P_{\boldsymbol{a}}^{\mathcal{G}}|u + v$ where $v < |P_{\boldsymbol{a}}^{\mathcal{G}}|$. Then we can write

$$\Lambda_{\mathcal{G}^{(t)},\boldsymbol{a}} = \left(\prod_{\boldsymbol{a} \in P_{\boldsymbol{a}}^{\mathcal{G}}} \lambda_{\mathcal{G},\boldsymbol{a}}\right)^{u} \lambda_{G,\mathcal{G}^0(\boldsymbol{a})} \ldots \lambda_{\mathcal{G},\mathcal{G}^v(\boldsymbol{a})}. \tag{3.5}$$

We see that circuit eigenvalues with this form allow precise estimation of the product of gate eigenvalues in the $\mathcal{G}$-orbit of $P_{\boldsymbol{a}}$. Indeed, this product can be estimated to *relative* precision by repeating the gate a number of times inversely proportional to the noise strength. By contrast, the individual gate eigenvalues within the orbit can only be estimated to *additive* precision.

Examining the covariance structure of the gate eigenvalue estimators in optimised ACES experimental designs reveals that ACES naturally exhibits this behaviour without modification. While the estimators for gate eigenvalues within an orbit have large variance, the covariance of these estimators ensures that the average over the orbit has small variance. This can be revealed by marginalising the gate eigenvalue estimator covariance matrix Σ over the gate orbits and ignoring parameters that can only be estimated to additive precision, such as those corresponding to state preparation and measurement (SPAM) noise. This yields the relative precision gate eigenvalue estimator covariance matrix Σ_R. Replacing Σ with Σ_R in the expression for the ordinary precision figure of merit $\mathcal{F}$ in [134] allows us to calculate the relative precision figure of merit $\mathcal{F}_R$.

In light of this, we make two changes to how we optimise ACES experimental designs. We optimise the depth of repeated circuits according to this relative precision figure of merit, whereas we still optimise the shallow circuits according to the ordinary precision figure of merit. The ordinary and relative precision figures of merit trade off against each other, which we manage by optimising the shot weights according to the product $\mathcal{F}\mathcal{F}_R$, or equivalently their geometric mean. This balances the relative increase in one figure of merit with the relative decrease in the other.

However, while ACES estimates the gate eigenvalues, and hence the gate error probabilities, to relative precision, this is not necessarily true after projecting the estimator into the probability simplex in the Euclidean distance. To preserve the estimator covariance information necessary for relative precision estimation, we instead project in the *Mahalanobis distance* [110], which weights the distance with the inverse of the gate probability estimator covariance matrix Σ_P^{-1}. This enables relative precision estimation by retaining estimator covariance information, and also improves the accuracy of the resulting estimator compared to Euclidean distance projection.

Let us first consider the inverse of the gate eigenvalue estimator covariance matrix Σ^{-1}, which for the generalised least squares estimator can be calculated as

$$\Sigma^{-1} = \operatorname{diag}(\boldsymbol{\lambda}^{-1}) A^{\intercal} \Omega'^{-1} A \operatorname{diag}(\boldsymbol{\lambda}^{-1}). \tag{3.6}$$

This calculation preserves the sparse structure of the design matrix A and the circuit log-eigenvalue estimator covariance matrix Ω'. We perform the calculation in Equation 3.6 using the gate eigenvalue estimator vector $\hat{\boldsymbol{\lambda}}$ and the estimator $\hat{\Omega}'$, whose elements can be calculated directly from measurement outcome data. Moreover, Ω' is sparse and block-diagonal, and applying the sparse Cholesky decomposition to each of its blocks allows us to invert it relatively easily.

Now consider concatenating the Walsh-Hadamard transform matrices, and their inverses, for each of the gates, which yields the block-diagonal Walsh-Hadamard transform matrix W, and its inverse W^{-1}. As W^{-1} maps the gate eigenvalues to the gate probabilities, we use its inverse to calculate

$$\Sigma_P^{-1} = W \Sigma^{-1} W^{\intercal}. \tag{3.7}$$

Our description of the transform matrices W and W^{-1} ignores an important subtlety. The gate eigenvalue estimator vector we work with omits the identity eigenvalues for each gate. Similarly, the gate probability estimator vector we project omits the identity probabilities for each gate. The identity eigenvalues and probabilities ensure normalisation of the probability distributions for each gate. We must omit these quantities to ensure that the corresponding covariance matrices have full rank. However, W and W^{-1} act on versions of these vectors with those omitted elements and must be conjugated by appropriate transform matrices, which can be determined straightforwardly, to reflect these omissions.

At large scales, generalised least squares becomes impractical and we resort to weighted least squares, as we did here when performing ACES on the syndrome extraction circuit of a distance-25 surface code. In this case, we only generate the diagonal of Ω', which can easily be inverted. Gate eigenvalue estimators are correlated within gate orbits because of the structure of the design matrix A, whereas the off-diagonal elements of Ω' result from the simultaneous estimation of circuit eigenvalues. Therefore, we expect that a version of Σ_P^{-1} calculated from only the diagonal of Ω' should suffice to ensure that Mahalanobis distance projection preserves the estimator covariance information necessary for relative precision estimation. Indeed, we find this to be the case in practice.

We perform Mahalanobis distance projection into the probability simplex with the convex solver SCS [137, 138]. While this can be performed for the entire gate probability estimator vector, this quickly becomes practically intractable. It suffices to perform the projection separately for each gate using its portion of the gate probability estimator vector and the corresponding block on the diagonal of Σ_P^{-1}, which only has a minor impact on performance.

Chapter 4

QuantumACES.jl: design noise characterisation experiments for quantum computers

> We should forget about small efficiencies, say about 97% of the time: premature optimization is the root of all evil.
>
> — Donald Knuth, *Structured Programming with* go to *Statements*

This chapter reproduces a forthcoming paper in the Journal of Open Source Software that briefly describes the open-source Julia package QuantumACES, and additionally includes an overview of QuantumACES in Section 4.A. The package consists of over 14,000 lines of optimised and documented Julia code, including more than 1,500 lines of tests that cover over 90% of the code in the package. QuantumACES makes it easy to generate the results in Chapter 2 and Chapter 3, as well as upcoming results in Chapter 5, merely on a laptop. I am quite proud of it.

I believe science should be open-source and replicable where possible. I developed QuantumACES with the aim of making a state-of-the-art scalable Pauli noise characterisation protocol openly accessible. Benchmarking quantum hardware, particularly in the key context of quantum error correction, should be free and easy. I hope QuantumACES will play some small role in the much larger story of fault-tolerant quantum computation.

Find QuantumACES at https://github.com/evanhockings/QuantumACES.jl, where you can also find its documentation.

4.1 Summary

QuantumACES is a Julia [108] package for designing, simulating, and performing scalable Pauli noise characterisation experiments for quantum computers. Noise in quantum devices is the key obstacle to large-scale quantum computation. Consequently, quantum computers will require fault-tolerant architectures that replace physical qubits and operations with redundantly encoded logical equivalents, which entails regularly measuring the parity checks of quantum error correcting codes [27–29]. Decoders process the resulting error syndrome data and attempt to infer the most likely underlying physical errors in the quantum device. Subsequently, they determine correction operations that

attempt to preserve logical information. The noise estimates produced by QuantumACES can inform decoders of the likelihood of physical error configurations in a quantum device [77, 81, 83, 84, 128], enabling noise-aware decoding. They can also be used to generate simulated data for training more accurate decoders, such as in [19], to verify appropriate device calibration, and to inform the co-design of quantum error correcting codes, decoders, fault-tolerant circuits, and quantum devices.

QuantumACES designs experiments to characterise Pauli noise in stabiliser circuits within the framework of averaged circuit eigenvalue sampling (ACES) [74], following the theory and protocol outlined in [134]. Stabiliser circuits are a restricted class of quantum circuits that admit efficient classical simulation [31, 32], including with Pauli noise. Quantum noise is tailored into Pauli noise by techniques such as Pauli frame randomisation [36], randomised compiling [37], or quantum error correction itself [38]. Additionally, the theory of quantum error correction and fault tolerance generally relies on modelling noise as Pauli noise [2].

QuantumACES contains routines for optimising designs for noise characterisation experiments, given an arbitrary stabiliser circuit and Pauli noise model, using functions that precisely predict the performance of these experimental designs. It has built-in circuits and noise models and also allows users to define their own. These noise characterisation experiments are simulated with the open-source Python package Stim [56], a fast simulator for stabiliser circuits with Pauli noise.

In a typical fault-tolerant quantum computing architecture, the bulk of the physical qubits and gate operations are dedicated to performing the syndrome extraction circuits that measure the parity checks of quantum error correcting codes. These syndrome extraction circuits, which are stabiliser circuits, are therefore the key target for noise characterisation experiments. QuantumACES is tailored to characterising Pauli noise in syndrome extraction circuits, particularly for topological quantum error correcting codes such as the surface code [5, 49–51]. It leverages the fact that the simple structures of the syndrome extraction circuits of topological quantum codes remain similar across code sizes. This enables optimised large-scale noise characterisation experiments that use experimental designs optimised at small scales. QuantumACES is capable of calculating and precisely fitting the performance scaling of these experimental designs as a function of the code size, enabling performance predictions at scales where explicit calculation becomes intractable.

Moreover, QuantumACES supports the simulation and decoding of memory experiments for syndrome extraction circuits with Stim and the open-source Python packages PyMatching [45, 57] and BeliefMatching [83], respectively. It also provides an interface with the open-source Python package Qiskit [139], enabling the export of experimental designs to Qiskit circuits that can then be implemented to characterise noise in real quantum devices.

4.2 Statement of need

The utility of detailed and scalable Pauli noise characterisation methods grows as experimental progress pushes quantum devices towards scales of hundreds of qubits and initial demonstrations of fault tolerance. QuantumACES enables noise characterisation and noise-aware decoding in this context, as demonstrated in [134] and [140], respectively. While there are several software packages for benchmarking and noise characterisation,

there are no open-source packages capable of detailed and scalable Pauli noise characterisation of quantum devices. Forest-Benchmarking [141] is an open-source Python package containing many routines for quantum characterisation, verification, and validation, but its detailed noise characterisation techniques are not scalable. Gate set tomography [62] is a principled and extremely detailed noise characterisation protocol implemented by the open-source Python package pyGSTi [142], but it is limited to characterising very small numbers of qubits. Cycle error reconstruction [76] is the noise characterisation protocol whose capabilities are most similar to ACES, but it is implemented by the commercial software True-Q [143].

Acknowledgements

This work was supported by the Australian Research Council Centre of Excellence for Engineered Quantum Systems (CE170100009), the U.S. Army Research Office (W911NF-21-1-0001, W911NF-23-S-0004), and the Unitary Foundation.

4.A Package overview

QuantumACES is a package for designing and simulating scalable and performant Pauli noise characterisation experiments for stabiliser circuits with averaged circuit eigenvalue sampling (ACES) [74]. It is focused on the context of quantum error correction and fault-tolerant circuits and, in particular, on the syndrome extraction circuits of topological quantum error correcting codes. It interfaces with Stim [56] for stabiliser circuit simulation, PyMatching [45, 57] and BeliefMatching [83] for decoding, and Qiskit [139] for implementation on quantum devices. QuantumACES uses PythonCall to interface with these Python packages in Julia.

Typical usage of QuantumACES involves first doing the following:

- Construct the circuit and the noise model that you aim to characterise, either using existing functions or your own.
- Optimise an ACES experimental design for noise characterisation of a small-scale instance of the circuit, typically according to a deterministic noise model, such as depolarising noise, with roughly the same average error rates as the noise you aim to characterise.

This subsequently enables:

- Transferring the optimised experimental design to larger-scale instances of the circuit, including with different noise models.
- Simulate noise characterisation experiments with ACES experimental designs, including at large scales, using Stim.
- Calculating performance predictions for experimental designs at small scales and fitting the performance predictions, in particular for syndrome extraction circuits as a function of the distance of the underlying code, to predict performance at large scales.
- Simulating memory experiments for syndrome extraction circuits using Stim, and then decoding with PyMatching or BeliefMatching with decoder priors informed by a range of noise models, including ACES noise estimates.
- Creating Pauli frame randomised ACES experimental designs, exporting them to Qiskit circuits, and processing the results, enabling implementation on quantum devices.

4.A.1 Example usage

QuantumACES is a registered Julia package and can be installed with the package manager. Once installed, it is easy to start using the package.

First, we create noise model parameters for depolarising noise and an instance of log-normal Pauli noise, a noise model described in Section 2.F, parameterised by the single-qubit gate infidelity `r_1`, two-qubit gate infidelity `r_2`, measurement infidelity `r_m`, and total standard deviation of the log-normal Pauli noise `sigma`, specifying the seed `seed` for reproducibility.

```
using QuantumACES
r_1 = 0.05 / 100
r_2 = 0.4 / 100
r_m = 0.8 / 100
total_std_log = 0.5
seed = UInt(0)
dep_param = get_dep_param(r_1, r_2, r_m)
log_param = get_log_param(r_1, r_2, r_m, total_std_log; seed = seed)
```

Similarly, we create circuit parameters for the syndrome extraction circuit of a distance `dist` rotated surface code.

```
dist = 3
rotated_param = get_rotated_param(dist)
```

Next, we create versions of this circuit with both noise models.

```
circuit_dep = get_circuit(rotated_param, dep_param)
circuit_log = get_circuit(rotated_param, log_param)
```

Then we optimise an ACES experimental design, controlling the optimisation by supplying keyword arguments to the type `OptimOptions`.

```
d = optimise_design(circuit_dep; options = OptimOptions(; seed = seed))
```

There are a number of options that can reduce the optimisation time. For example, we can disable cyclic coordinate descent optimisation of the circuit depth of repeated tuples in the design by setting `max_cycles = 0`. We can also allow the greedy search over ordinary tuples to terminate once they are left unchanged by single excursion in the search by setting `excursions_unchanged = 1`.

```
d = optimise_design(
    circuit_dep;
    options = OptimOptions(;
        max_cycles = 0,
        excursions_unchanged = 1,
        seed = seed,
    )
)
```

This experimental design can then be transferred to the circuit with the log-normal Pauli noise model.

```
d_log = generate_design(circuit_log, d)
```

If we only wish to update the noise model, however, we can do this more efficiently.

```
d_log = update_noise(d, circuit_log)
```

Now we can simulate `repetitions` instances of ACES noise characterisation across a range of measurement budgets `budget_set`, leveraging Stim to perform the simulation.

```
budget_set = [10^6; 10^7; 10^8]
repetitions = 1000
aces_data = simulate_aces(d_log, budget_set;
    repetitions = repetitions,
    seed = seed,
)
```

We can compare the noise estimates to performance predictions and print the z-scores.

```
merit_log = calc_merit(d_log)
pretty_print(aces_data, merit_log)
```

Code similar to this can be used to generate the data in Figure 2.3.

We can also simulate ACES noise characterisation at scale. First, create a new design at a large code distance `dist_big`. Setting `full_covariance = false` means only the diagonal circuit eigenvalue estimator covariance matrix is generated, which saves a substantial amount of time. It also prevents the design from attempting to perform generalised least squares (GLS) with the full covariance matrix, which can consume large amounts of memory at large scales, restricting the design to weighted least squares (WLS).

```
dist_big = 25
rotated_param_big = get_rotated_param(dist_big)
circuit_big = get_circuit(rotated_param_big, dep_param)
circuit_big_log = get_circuit(rotated_param_big, log_param)
d_big = generate_design(circuit_big_log, d;
    full_covariance = false,
    diagnostics = true,
)
```

Now simulate this new design, setting `split = true` to avoid memory issues.

```
aces_data_big = simulate_aces(d_big, budget_set; split = true, seed = seed)
```

Code similar to this can be used to generate the data in Figure 2.7 and Figure 2.8.

It is expensive to directly calculate the performance of the experimental design at this scale. Instead, we calculate the performance scaling of the experimental design at small code distances and then extrapolate. We can do this for depolarising noise, and for an average over instances of log-normal Pauli noise, calculating up to `dist_max`, and then extracting fits.

```
dist_max = 11
merit_scaling = calc_merit_scaling(d, dist_max)
scaling_fit = get_scaling_fit(merit_scaling)
ensemble_scaling = calc_ensemble_scaling(d_log, dist_max; seed = seed)
ensemble_fit = get_ensemble_fit(ensemble_scaling)
```

Code similar to this can be used to generate the data in Figure 2.4.

This allows us to predict expectations and variances, and compare them to the true values, for both the ordinary figure of merit and the relative precision figure of merit.

```
wls_pred_expectation = ensemble_fit.wls_expectation_model(dist_big)
wls_pred_variance = ensemble_fit.wls_variance_model(dist_big)
wls_scores_big = [
    (noise_error.wls_nrmse .- wls_pred_expectation) / sqrt(wls_pred_variance)
    for noise_error in aces_data_big.noise_error_coll[1, :]
]
wls_pred_relative_expectation =
    ensemble_fit.wls_relative_expectation_model(dist_big)
wls_pred_relative_variance =
    ensemble_fit.wls_relative_variance_model(dist_big)
wls_relative_scores_big = [
```

```
    (noise_error.wls_relative_nrmse .- wls_pred_relative_expectation) /
    sqrt(wls_pred_relative_variance)
    for noise_error in aces_data_big.noise_error_coll[1, :]
]
```

These are not exactly z-scores in particular because the simulation was for a single instance of log-normal Pauli noise, whereas the predictions are averaged over instances. In practice, performance appears to be self-averaging so prediction works well at scale.

We can now use Stim to simulate a memory experiment with `big_rounds` rounds, sampling `big_shots` shots. We inform the decoder, PyMatching by default, with a range of noise models including our noise estimates.

```
big_rounds = dist_big
big_shots = 5 * 10^6
decoder_gate_probabilities = [
    circuit_big_log.gate_probabilities
    circuit_big.gate_probabilities
    [noise_est.wls_gate_probabilities
    for noise_est in aces_data_big.noise_est_coll[1, :]
    ]
]
decoder_labels = [
    "True"
    "Depolarising"
    ["ACES S=$(budget)" for budget in budget_set]
]
big_memory_data = simulate_memory(circuit_big_log, big_rounds, big_shots;
    seed = seed,
    decoder_gate_probabilities = decoder_gate_probabilities,
    decoder_labels = decoder_labels,
    diagnostics = true,
)
big_memory_summary = get_memory_summary(big_memory_data)
```

Code similar to this can be used to generate the data in Table 3.1.

We can also construct a Pauli frame randomised version of the experimental design and generate corresponding Qiskit circuits. We specify a minimum number of randomisations `min_randomisations` and a target shot budget `target_shot_budget`, and `experiment_shots` shots per randomised experiment.

```
min_randomisations = 128
target_shot_budget = 5 * 10^6
experiment_shots = 64
d_rand = generate_rand_design(
    d_log,
    min_randomisations,
    target_shot_budget,
    experiment_shots;
    seed = seed,
)
```

This modifies the shot weights, so we can calculate the merit of this new design.

```
d_shot = get_design(d_rand)
merit_shot = calc_merit(d_shot)
```

Now we simultaneously generate ensembles of Stim and Qiskit circuits that implement this experimental design. The Qiskit circuits act on `qiskit_qubit_num` qubits and `qiskit_qubit_map` maps QuantumACES qubit indices to Qiskit qubit indices, noting that Julia indexes from 1 whereas Python indexes from 0.

```
qiskit_qubit_num = 17
qiskit_qubit_map = collect(0:(qiskit_qubit_num - 1))
(stim_ensemble, qiskit_ensemble) =
    get_stim_qiskit_ensemble(d_rand, qiskit_qubit_num, qiskit_qubit_map)
```

We only simulate in Stim as the Qiskit stabiliser circuit simulator is much slower.

```
simulate_stim_ensemble(d_rand, stim_ensemble, experiment_shots; seed = seed)
rand_noise_est = estimate_stim_ensemble(d_rand, experiment_shots;
    simulation_seed = seed)
rand_noise_error = get_noise_error(d_rand, rand_noise_est)
rand_noise_score = get_noise_score(rand_noise_error, merit_shot)
```

Suppose we then run the Qiskit circuits on a quantum device. The results must be stored in an appropriate folder to be processed. Given a prefix `backend`, which typically describes the device on which the circuits are run, the results must be stored relative to the current directory in a folder whose name is given by `qiskit_results_folder`.

```
backend = "backend"
d_rand_filename = rand_design_filename(d_rand)
@assert d_rand_filename[(end - 4):end] == ".jld2"
qiskit_results_folder = "data/$(backend)_$(d_rand_filename[1:(end - 5)])"
```

The ensemble `qiskit_ensemble` is a vector containing vectors of Qiskit circuits, each of which comprise a job. Each job should be stored as a pickle in `qiskit_results_folder` with prefix `prefix`, followed by an underscore and the job index, starting from 1.

```
prefix = "job"
example_job_1_filename = "$(qiskit_results_folder)/$(prefix)_1.pickle"
```

Then it is simple to process the data and estimate the noise.

```
process_qiskit_ensemble(
    d_rand,
    qiskit_qubit_num,
    qiskit_qubit_map,
    experiment_shots;
    backend = backend,
    prefix = prefix,
)
noise_est = estimate_qiskit_ensemble(
    d_rand,
    qiskit_qubit_map,
    experiment_shots;
    backend=backend,
)
```

Finally, we can analyse the consistency of this noise estimate with our noise model.

```
model_violation = get_model_violation(d_shot, noise_est)
```

This quantity is a z-score which is approximately normally distributed if the circuit-level Pauli noise model is upheld, as it is in simulation. Note the model violation score is substantially larger when calculated for the projected noise estimates, that is, the noise estimates after projecting the Pauli error probabilities into the probability simplex even in simulation. By default, then, this function calculates the model violation for the unprojected noise estimates.

We can also create a version of the experimental design corresponding to a combined noise model for which Pauli X, Z, and Y basis SPAM noise are combined into a single parameter for each qubit, so that for n qubits we have n SPAM noise parameters. Previously, we considered $3n$ SPAM noise parameters for each Pauli basis, which we will call the ordinary noise model. Then we can estimate the noise with the combined noise model and calculate its model violation.

```
d_comb = get_combined_design(d_shot)
comb_noise_est = estimate_gate_noise(d_comb, noise_est)
comb_model_violation = get_model_violation(d_comb, comb_noise_est)
```

We might want to perform model selection with the Akaike information criterion (AIC) or the Bayesian information criterion (BIC), which are straightforward to calculate.

```
aic = get_aic(d_shot, noise_est)
bic = get_bic(d_shot, noise_est)
comb_aic = get_aic(d_comb, comb_noise_est)
comb_bic = get_bic(d_comb, comb_noise_est)
```

The preferred model is the one that minimises the AIC or BIC, depending on the metric of choice. In practice, the combined noise model tends not to be formally preferred but is nevertheless more parsimonious as the SPAM noise estimates in the ordinary noise model differ across Pauli bases more than can reasonably be expected. Therefore we tend not to use this model selection procedure.

4.A.2 Creating circuits and noise models

QuantumACES makes it easy to create new circuits and noise models. At a high level, we create new parameter types for the circuit or noise model and then create new methods for the functions `get_circuit` and `init_gate_probabilities`, respectively, that take these parameter types as arguments. Then `get_circuit` uses the circuit and noise model parameters to create a `Circuit` object.

The `Circuit` object enables the functionality we saw in Section 4.A.1, with two exceptions. First, calculating performance predictions requires the circuit to be parameterised by a `dist` parameter, typically the distance of the code underlying the syndrome extraction circuit. Second, simulating memory experiments in Stim requires the circuit to be a syndrome extraction annotated with the appropriate information. The `Circuit` object contains an `extra_fields` field which is a dictionary that can store additional parameters to enable functionality such as this. In particular, syndrome extraction circuits must store a `CodeParameters` object in this dictionary, which is then used to generate the detectors in Stim for memory experiment circuits, enabling decoding.

As an example, we create the example circuit in Figure 2.2.

```
struct ExampleParameters <: AbstractCircuitParameters
    params::Dict{Symbol, Any}
    circuit_name::String
end
```

The parameters for the circuit are stored in the dictionary `params`. We would then create a constructor for this type, which would initialise the required parameters in the dictionary `params`, but omit the details here. As an example, the syndrome extraction circuits of codes such as the surface code store the vertical and horizontal distances of the code, amongst other parameters, in their `params` field. Then we create a function to create the circuit from the parameters.

```
function example_circuit(example_param::ExampleParameters)
    # Set up variables
    pad_identity = example_param.params[:pad_identity]
    layer_time_dict = example_param.params[:layer_time_dict]
    single_qubit_type = :single_qubit
    two_qubit_type = :two_qubit
    # Generate the circuit and data
    qubit_num = 3
    circuit = [
        Layer([Gate("CZ", 0, [2; 3])], qubit_num),
        Layer([Gate("CZ", 0, [1; 2]), Gate("H", 0, [3])], qubit_num),
        Layer([Gate("H", 0, [1]), Gate("S", 0, [2]), Gate("H", 0, [3])],
            qubit_num),
    ]
    layer_types = [two_qubit_type, two_qubit_type, single_qubit_type]
    layer_times = get_layer_times(layer_types, layer_time_dict)
    extra_fields = Dict{Symbol, Any}()
    # Pad layers with identity gates if appropriate
    if pad_identity
        circuit = [pad_layer(l) for l in circuit]
    end
    return (
        circuit::Vector{Layer},
        layer_types::Vector{Symbol},
        layer_times::Vector{Float64},
        extra_fields::Dict{Symbol, Any},
    )
end
```

Finally, we add a method to `get_circuit`, using the following form, to create the circuit from the parameters.

```
function QuantumACES.get_circuit(
    example_param::ExampleParameters,
    noise_param::T;
    noisy_prep::Bool = false,
    noisy_meas::Bool = true,
    combined::Bool = haskey(noise_param.params, :combined) ?
        noise_param.params[:combined] : false,
    strict::Bool = false,
) where {T <: AbstractNoiseParameters}
    (circuit, layer_types, layer_times, extra_fields) =
        example_circuit(example_param)
    c = get_circuit(
        circuit,
        layer_types,
        layer_times,
```

```
        noise_param;
        circuit_param = example_param,
        extra_fields = extra_fields,
        noisy_prep = noisy_prep,
        noisy_meas = noisy_meas,
        combined = combined,
        strict = strict,
    )
    return c::Circuit
end
```

As another example, we consider creating a phenomenological noise model.

```
struct PhenomenologicalParameters <: AbstractNoiseParameters
    params::Dict{Symbol, Any}
    noise_name::String
end
```

Again, we would then create a constructor for this type, but omit the details here. Now we add a method to `init_gate_probabilities` to create the Pauli error probabilities for each gate from the parameters.

```
function QuantumACES.init_gate_probabilities(
    total_gates::Vector{Gate},
    phen_param::PhenomenologicalParameters,
)
    # Set up variables
    p = phen_param.params[:p]
    m = phen_param.params[:m]
    m_r = phen_param.params[:m_r]
    m_i = phen_param.params[:m_i]
    # Determine the weights of the Pauli errors
    one_qubit_support_size = ones(3)
    n = 2
    two_qubit_support_size = Vector{Int}()
    bit_array = BitArray(undef, 2n + 1)
    for bit_array.chunks[1] in 1:(4^n - 1)
        two_qubit_pauli = Pauli(convert(Vector{Bool}, bit_array), n)
        push!(two_qubit_support_size, length(get_support(two_qubit_pauli)))
    end
    @assert sum(two_qubit_support_size .== 1) == 6
    @assert sum(two_qubit_support_size .== 2) == 9
    @assert length(two_qubit_support_size) == 15
    # Generate the noise
    gate_probabilities = Dict{Gate, Vector{Float64}}()
    for gate in total_gates
        if is_spam(gate)
            gate_probs = [m]
        elseif is_mid_meas_reset(gate)
            gate_probs = [m_r]
        elseif is_meas_idle(gate)
            gate_probs = m_i .^ one_qubit_support_size
        else
            gate_support_size = length(gate.targets)
            if gate_support_size == 1
```

```
                gate_probs = p .^ one_qubit_support_size
            elseif gate_support_size == 2
                gate_probs = p .^ two_qubit_support_size
            else
                throw(error("The gate $(gate) is unsupported."))
            end
        end
        @assert sum(gate_probs) < 1
        gate_probabilities[gate] = [1 - sum(gate_probs); gate_probs]
    end
    return gate_probabilities::Dict{Gate, Vector{Float64}}
end
```

Once these two methods have been defined, it becomes easy to create the example circuit in Figure 2.2 with a phenomenological noise model.

```
circuit_example = get_circuit(example_param, phen_param)
```

This enables the relevant functionality in Section 4.A.1.

4.A.3 Package performance

QuantumACES has several performance optimisations that are essential for its practical utility. We now examine some core optimisations that may be of particular interest through the lens of the functions `optimise_design` and `simulate_aces`. These functions optimise experimental designs and simulate ACES noise characterisation experiments, respectively.

First, we discuss the optimisation of experimental designs with `optimise_design`. Crucial to the tractability of this optimisation is the fact that we can perform the optimisation at small scales, such as the syndrome extraction circuits of distance-3 topological codes, and then transfer the optimised design to the syndrome extraction circuits of the same codes at larger distances, while retaining performance characteristics. As we discussed in Section 2.3.5, an ACES experimental design $(\mathcal{T}, \Gamma)$ is parameterised by the tuple set $\mathcal{T}$ and the shot weights Γ. The tuple set $\mathcal{T}$ is discrete and cannot be optimised continuously, and so its optimisation relies on repeated calculation of the figure of merit $\mathcal{F}$, which is given in Equation 2.28. However, the shot weights Γ are continuous, allowing us to optimise them with gradient descent. This relies on the fast calculation of the gradient of the figure of merit with respect to the shot weights, with analytic expressions for these gradients given in Section 2.D. Carefully optimised calculation of these analytic expressions is roughly one and a half orders of magnitude faster than forward-mode automatic differentiation, which is much faster than backward-mode automatic differentiation in this case.

We now remark on some considerations in the linear algebra calculations of the figure of merit and its gradient. The design matrix A is sparse, and the circuit (log-)eigenvalue estimator covariance matrix Ω (Ω') is block diagonal with the blocks themselves being sparse. It is important to appropriately leverage this sparsity in calculations. Matrix multiplications quickly densify sparse matrices, at which point the matrices must be converted to dense form to avoid substantial performance degradation. In general, QuantumACES calculates matrix inverses for dense symmetric matrices, and so leverages the Cholesky or Bunch-Kaufman decompositions depending on whether the matrix is positive definite or not, respectively. It is often necessary to invert Ω or Ω', and this

leverages the sparse Cholesky decomposition for separate blocks of the matrix. Lastly, QuantumACES calculates the condition number and pseudoinverse norm of A with sparse methods for eigenvalue finding.

Next, we discuss the simulation of ACES noise characterisation experiments with `simulate_aces`. Experiments are simulated with Stim [56], a fast simulator for stabiliser circuits with Pauli noise, with the function `simulate_stim_estimate`. This function simulates experiments with Stim and immediately estimates the circuit eigenvalues measured by the experiment to avoid storing large numbers of measurement outcomes in memory. By default, Stim outputs measurement outcomes as bits, but can instead pack measurement outcomes into 8-bit unsigned integers, which in practice speeds up sampling of circuit outcomes by roughly a factor of 8. These bit-packed results can then be processed quickly with careful bit manipulation to estimate the circuit eigenvalues measured by the experiment. It is also important to automatically split up Stim simulations that attempt to sample too many shots at once to avoid memory issues. One step in this simulation process that can surprisingly become a bottleneck is the creation of the circuit string for Stim corresponding to each experiment. If the string manipulation is not optimised, this can become very slow for large circuits.

Once the Stim simulations have been performed to estimate the circuit eigenvalues, we then estimate the gate eigenvalues and Pauli error probabilities with the function `estimate_gate_noise`. We will discuss this process in the context of GLS, though at large scales it is essential to only construct the diagonal of the circuit eigenvalue estimator covariance matrix Ω and hence only perform WLS. First, we calculate the sparse block diagonal Cholesky factorisation of the circuit log-eigenvalue estimator covariance matrix $\Omega' = LL^\intercal$, and calculate the inverse of the Cholesky factor L^{-1}. Then we left-divide the scaled design matrix $L^{-1}A$ by the scaled circuit log-eigenvalues $L^{-1}\boldsymbol{b}$ to obtain the estimated gate log-eigenvalues $\hat{\boldsymbol{x}}$. The left-division operator in Julia leverages the sparse structure of the inversion problem and is very fast. Once we have the gate log-eigenvalues, we can straightforwardly estimate the Pauli error probabilities for each gate in the circuit.

However, these estimates are not guaranteed to be valid probability distributions, so we must project the estimates into the probability simplex. As discussed in Section 3.A, we perform this projection in the Mahalanobis distance with the fast convex solver SCS [137, 138]. In particular, we use Equation 3.6 to optimise the calculation of the precision matrix such that the only matrix inversion required is the highly-optimised inversion of the circuit log-eigenvalue estimator covariance matrix Ω'.

These core optimisations make QuantumACES capable of optimising ACES experimental designs and simulating ACES noise characterisation experiments at large scales using only a laptop. In particular, the results presented in this thesis were produced on a 2021 M1 Max Macbook Pro with 32 GB of RAM. I believe QuantumACES provides a proof of concept demonstrating the utility and practicality of scalable circuit-level Pauli noise characterisation in the context of quantum error correction and fault-tolerant quantum computation.

Chapter 5

A heavy hexagon memory with noise-aware decoding

> To ask in advance for a complete recipe would be unreasonable. We can specify only the human qualities required: patience, flexibility, intelligence.
>
> — John von Neumann, *Can We Survive Technology?*

The following chapter describes ongoing work in which we experimentally implement the methods described in Chapter 2 and Chapter 3 on IBM quantum devices, using the software package QuantumACES described in Chapter 4. We perform noise characterisation of the syndrome extraction circuit of the heavy hexagon code, examine the consistency of our noise estimates, and use the results to design an improved syndrome extraction circuit. Then we operate the heavy hexagon code as a quantum memory and use the noise estimates to predict the performance of the memory and perform noise-aware decoding. I believe we will be able to improve quite significantly on the preliminary results presented here.

5.1 Introduction

Until now we have worked only in theory and simulation. But noise is ultimately a phenomenon observed in physical quantum devices. So it is now time to get our hands dirty and implement these methods experimentally on real quantum hardware.

We will perform our experiments on IBM quantum devices, high-performance superconducting quantum devices with low error rates and over a hundred qubits. The qubits are arranged in a heavy hexagon lattice, a topology that is the product of a co-design process [144]. Heavy hexagon codes are a family of subsystem codes that are well-suited to this topology [52]. In the heavy hexagon lattice, most qubits have degree-2 connectivity while the rest have degree-3 connectivity. This makes manufacturing high-performance devices easier, as it reduces gate frequency collisions in the IBM superconducting hardware platform [52]. Previous work has examined the heavy hexagon code operated as a quantum memory [77].

In this chapter, we focus on noise characterisation and noise-aware decoding for the heavy hexagon code. First, in Section 5.2 we describe heavy hexagon codes and design a novel syndrome extraction circuit optimised for the noise and device characteristics of the

IBM Heron quantum architecture. Then Section 5.3 outlines the details of experimentally implementing the scalable Pauli noise characterisation protocol introduced in Chapter 2 based on averaged circuit eigenvalue sampling (ACES), performed with the software described in Chapter 4. Following this, Section 5.4 examines the model error associated with our experimental noise estimates. Finally, Section 5.5 presents our heavy hexagon memory results, including performance predictions derived from our noise characterisation results, and the performance of noise-aware decoding, following the methods of Chapter 3. We close with a discussion of our results and future directions in Section 5.6.

5.2 Heavy hexagon codes

Heavy hexagon codes are a family of subsystem codes co-designed with the heavy hexagon lattice. We introduced surface codes and briefly mentioned heavy hexagon codes in Section 1.3.3. In this section, we describe heavy hexagon codes, which can be thought of as modified surface codes, and a standard syndrome extraction circuit. We also introduce a novel syndrome extraction circuit whose design is informed by ACES noise estimates and the characteristics of IBM quantum devices.

A *heavy hexagon code* is a subsystem code that encodes a single logical qubit in a $d_Z \times d_X$ rectangular array of data qubits. We usually consider square codes with $d = d_Z = d_X$. These codes have an asymmetry between the X and Z gauge generators, but we are free to exchange the Pauli types of the gauge generators. Here we follow the convention of [52], whereas [77] uses the opposite convention.

In heavy hexagon codes, the X gauge generators are exactly the same as the surface code X stabiliser generators, namely, weight-4 products of Pauli X on plaquettes in the array, and weight-2 products on the boundary. However, the Z gauge generators differ from the surface code Z stabiliser generators. Each weight-4 surface code Z stabiliser generator is split into two weight-2 heavy hexagon code Z gauge generators, which are products of Pauli Z on edges in the array along the logical X direction. Like in the surface code, the logical Pauli Z (X) generator can be chosen to be any straight string of Pauli Z (X) operators of length d_Z (d_X) that terminates in a boundary Z (X) gauge generator, which is checked by the X (Z) gauge generators. Figure 5.1 shows a syndrome extraction circuit for a $d = 3$ heavy hexagon code following the circuit and Pauli type convention given in [77], which is opposite to the description here.

The advantage of subsystem codes such as heavy hexagon codes is that syndrome extraction circuits can use lower-degree connectivity measure qubits. In Figure 5.1, the weight-2 type checks, now X instead of Z, are performed by X measure qubits connected to pairs of data qubits. The weight-4 type checks, now Z instead of X, are performed by Z measure qubits connected to paids of X measure qubits. Hence the Z measure qubits, and those X measure qubits not connected to Z measure qubits are degree-2, whereas the X measure qubits connected to Z measure qubits are degree-3. In contrast, the measure qubits in the surface code are all degree-4.

The stabiliser group for this code differs from the surface code because the Z gauge generators now anticommute with the X gauge generators. In fact, the Z stabiliser generators of the heavy hexagon code are exactly the Z stabiliser generators of the surface code. However, the X stabiliser generators are products of the X gauge generators along the logical X direction. Hence the code has a threshold for X errors but does not have a threshold for Z errors. This is because the size of the X stabiliser generators

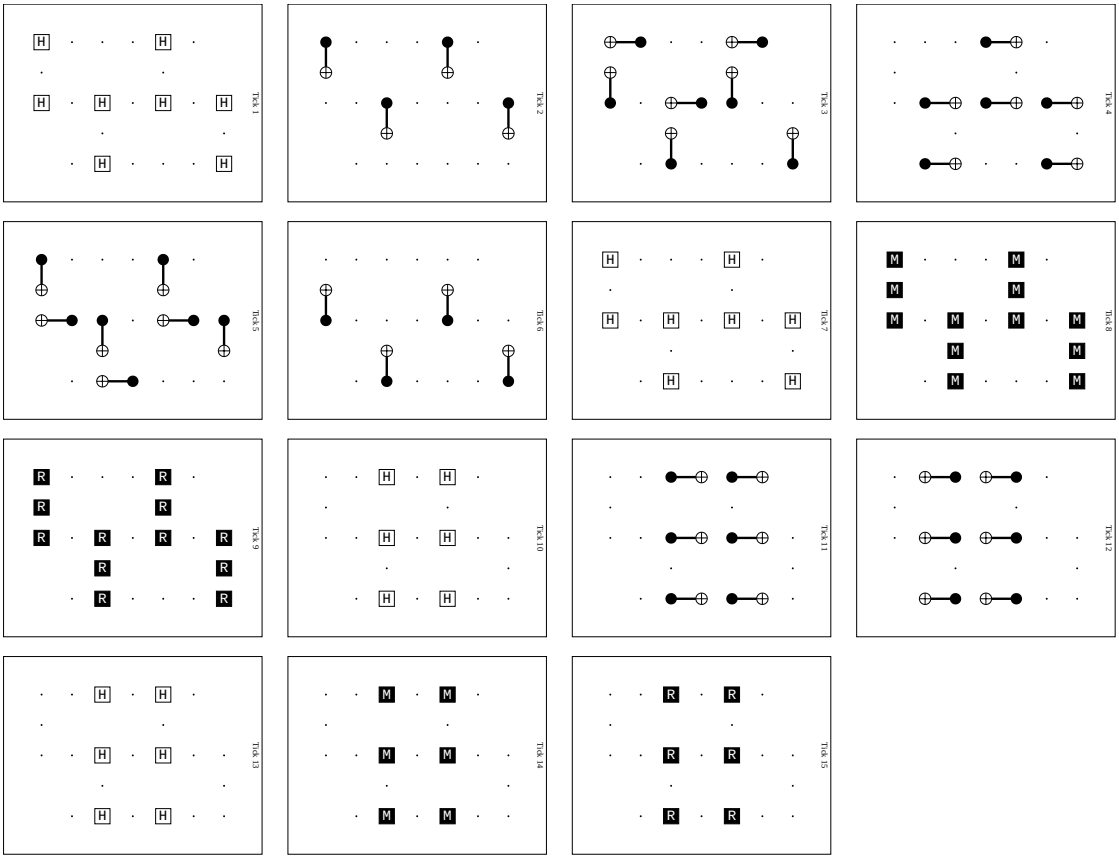

Figure 5.1: Diagram of a syndrome extraction circuit for a distance-3 heavy hexagon code, following the circuit given in [77], displayed with Stim [56].

grows with the logical X distance d_X, and so are flipped by an increasingly large number of Z errors as d_X grows.

Now we turn our attention to syndrome extraction circuits for the heavy hexagon code. We performed preliminary ACES noise characterisation experiments on the circuit in Figure 5.1 in both IBM Sherbrooke, an older processor based on the IBM Eagle architecture, and IBM Fez, a new processor based on the IBM Heron architecture. In IBM Sherbrooke, we observed extreme amounts of noise on the idling qubits during the controlled-X gate layers, particularly in the form of Pauli Z errors. Indeed, ZZ crosstalk with idling qubits during two-qubit gates is a known issue with the IBM Eagle architecture. It was encouraging to see that we had similar findings, particularly because this correlated error mechanism is not explicitly represented in our noise model. This demonstrates that ACES is indeed capable of detecting the averaged effect of correlated errors, as we argued in Section 2.A.

We did not observe this crosstalk issue in IBM Fez, whose newer IBM Heron architecture was designed to eliminate this issue. While we found that the gates in IBM Fez perform very well, noise on idling qubits during measurement was very substantial. This fact becomes unsurprising when we consider the time taken for two-qubit gates and measurements in IBM Fez. These times are publicly displayed on the IBM Quantum website, though they are subject to change as devices are calibrated and improved. Currently, in IBM Fez two-qubit gates take 84 ns and measurements take 1560 ns, so

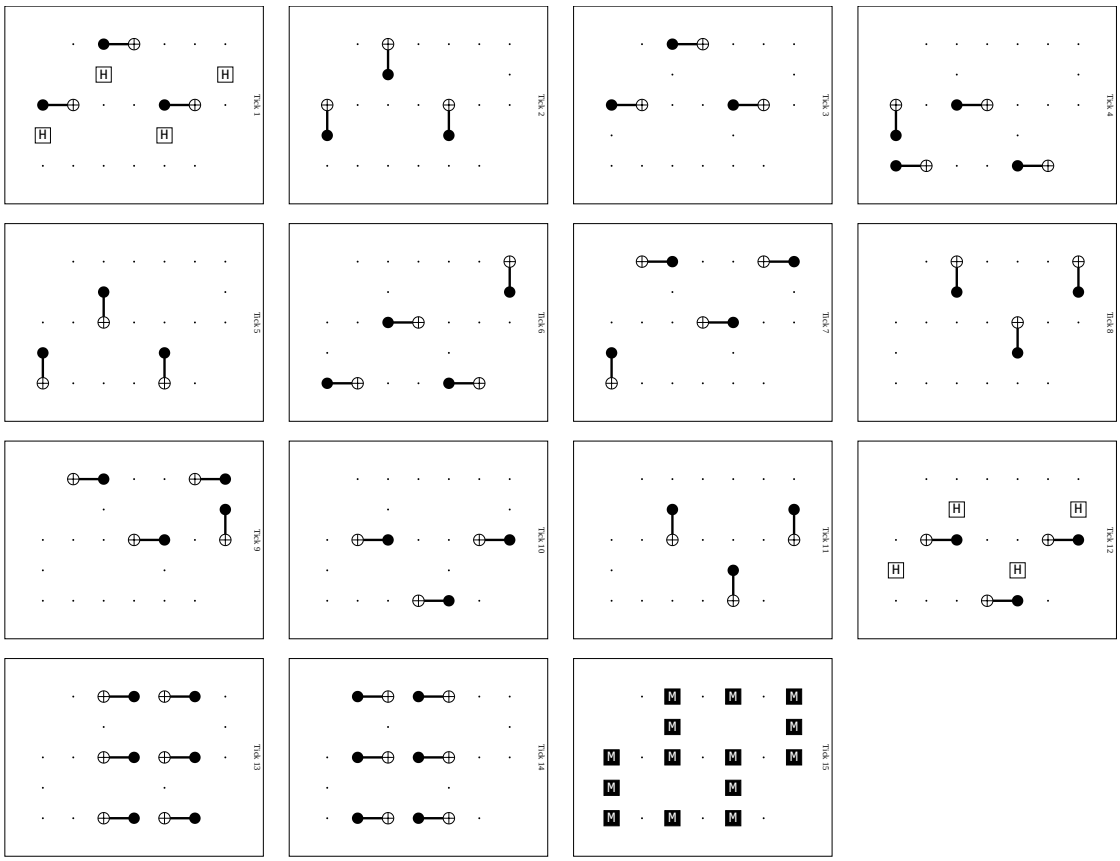

Figure 5.2: Diagram of an improved syndrome extraction circuit for a distance-3 heavy hexagon code displayed with Stim [56]. Compared to Figure 5.1, the qubit layout is flipped horizontally, and the circuit features twice as many two-qubit gate layers but only a single measurement layer.

measurements take about 18.6 times longer than two-qubit gates. For comparison, in IBM Sherbrooke two-qubit gates take 533 ns and measurements take 1216 ns, so measurements only take about 2.3 times longer than two-qubit gates. The syndrome extraction circuit in Figure 5.1 features 7 two-qubit gate layers and two measurement layers in each round of syndrome extraction, as it measures the X and Z stabilisers separately. This is a reasonable circuit design for IBM Sherbrooke, but it is not ideal for IBM Fez, where we would prefer only a single measurement layer.

Accordingly, we designed a new syndrome extraction circuit for the heavy hexagon code with only a single measurement layer, shown in Figure 5.2. Compared to Figure 5.1, the qubit layout is flipped horizontally, which does not alter the code, and the circuit returns to the X and Z stabiliser types of [52]. This circuit features 14 two-qubit gate layers, which is twice as many as the original syndrome extraction circuit, but only a single measurement layer. Adding 7 two-qubit gate layers to remove a single measurement layer reduces the time taken to perform a round of syndrome extraction in IBM Fez—though not IBM Sherbrooke—so we expect this circuit will reduce idling errors. This syndrome extraction circuit for the heavy hexagon code is the design we will consider in the remainder of this chapter.

Another important consideration is whether to reset the measure qubits after measure-

ment. While the syndrome extraction circuit in Figure 5.1 includes reset operations, the circuit in Figure 5.2 does not. IBM devices are not capable of performing unconditional resets, which use a physical reset operation to return a qubit to the $|0\rangle$ state. Instead, they perform conditional resets, which consist of a computational basis measurement followed by a Pauli X conditioned on the measurement outcome indicating the qubit was in the $|1\rangle$ state. This is simply a Pauli correction that could be tracked in software, and it has been shown that conditional resets are no better than measurement [145]. Now suppose the circuit in Figure 5.2 included conditional reset. Measurement takes longer than all of the gate layers in the circuit combined, so repeating the version with conditional reset twice would take longer than repeating the actual circuit three times. Hence we do not include conditional reset.

The absence of unconditional reset is relatively well-tolerated in memory experiments [145], but is more harmful when performing logical operations. This is better tested in stability experiments [146], which investigate moving a logical observable through space, rather than in memory experiments, which investigate preserving a logical observable through time. In memory experiments, logical errors are caused by spacelike strings of physical errors across the code, whereas in stability experiments, logical errors are caused by timelike strings of measurement errors. Without unconditional reset, each measurement error also causes an erroneous misclassification of the measurement in the next round, halving the timelike code distance. However, this chapter focuses on memory experiments for which this is not a major issue.

The circuit in Figure 5.2 is the product of a co-design process and reflects a focus on quantum error correction circuits over quantum error correcting codes, as I discussed in Section 1.3.4. We have designed the syndrome extraction circuit to leverage the strengths of the IBM Heron architecture while avoiding its weaknesses, and this process was aided by noise characterisation results. This co-design loop could be closed by measurement-focused hardware improvements.

5.3 Implementing ACES experimentally

Experimentally implementing the ACES protocol described in Chapter 2 for IBM quantum devices performing the syndrome extraction circuit in Figure 5.2 is largely straightforward. At a high level, we simply need to optimise an experimental design for this circuit, generate Qiskit circuits that implement the experimental design, run those circuits on the device, and process the data. The primary challenge in this experimental implementation is the need to perform Pauli frame randomisation or randomised compiling to tailor the noise into Pauli noise, something that was unnecessary in simulations. This section describes how we address this challenge.

Ideally, we would use a separate randomisation each time we sample a shot from a circuit on a quantum device. It has been shown in the context of a related protocol, shadow estimation, that sampling multiple shots from a single randomisation is maximally ineffective for Clifford circuits [147]. Unfortunately, the syndrome extraction circuits we consider here are Clifford circuits, so we would like to minimise the number of shots we sample from each randomisation.

However, the control systems of superconducting quantum devices, such as those produced by IBM, use waveform generators to produce the pulse sequences that perform quantum circuits on the device [148]. Loading these pulse sequences onto the device

takes a significant amount of time. Hence, given a fixed shot budget, increasing the number of randomisations also increases the runtime of the protocol. While a method for hardware-efficient randomised compiling has recently been introduced [101], which randomises each shot with little or no overhead, these methods must be implemented in the control system of the device. They are not currently supported by IBM quantum devices, and we do not have control system access. Regardless, such modifications are outside the scope of this work.

We must manage the runtime of the protocol by carefully choosing the number of randomisations. We will do so in a way that approximates the shot weights first introduced in Section 2.3.3. Suppose we have an experimental design $(\mathcal{T}, \Gamma)$ with tuple set $\mathcal{T}$ and shot weights Γ, as well as a target number of measurement shots S', also called the shot budget. Given the *shots per randomisation* S'_R, we aim to determine the randomisations $\mathcal{R} = \{R_T\}_{T\in\mathcal{T}}$ for each tuple in the tuple set.

Recall from Section 2.3.3 that each tuple $T \in \mathcal{T}$ is allocated a fraction Γ_T of the shots. These shots are divided evenly between the experiments in its experiment set $\mathcal{E}_T$. Suppose that we perform R_T randomisations of each experiment in $\mathcal{E}_T$, then we can see that

$$\Gamma_T = \frac{R_T|\mathcal{E}_T|}{\sum_{T\in\mathcal{T}} R_T|\mathcal{E}_T|}. \tag{5.1}$$

Thus we aim to choose the randomisations $\mathcal{R}$ in order to best approximate the shot weights Γ, and do so algorithmically.

First, initialise the randomisations $\mathcal{R}$ to a minimum value R_{min}. Then, given the shots per randomisation S'_R and a target number of measurement shots S', also known as the shot budget, we iteratively increment the randomisations towards better approximating the shot weights. At each step, we increment by one the number of randomisations of one tuple, greedily choosing this tuple to minimise the 2-norm between the shot weights Γ and the shot weights currently entailed by the randomisations. We terminate once the target shot budget is reached, giving us the randomisations $\mathcal{R}$.

When initially implementing our ACES noise characterisation protocol, we used $S'_R = 4096$ shots per randomisation. After realising the importance of minimising this value, we reduced this to $S'_R = 500$. Then, leveraging control system improvements that were made for the IBM quantum devices, we were able to further reduce this to $S'_R = 64$. Ultimately, this allowed us to gather $S' = 5 \times 10^6$ shots in a little over an hour on IBM Fez. We present the full noise characterisation results and the experimental design used to obtain them in Section 5.A.

These results are shown for a *combined* noise model for which Pauli X, Z, and Y basis SPAM noise are combined into a single parameter for each qubit, so that for n qubits we have n SPAM noise parameters. In Chapter 2, we considered $3n$ SPAM noise parameters for each Pauli basis, and we will call this the *ordinary* noise model. The combined noise model is more parsimonious as the SPAM noise estimates for the ordinary noise model differ across Pauli bases more than can reasonably be expected given the small number of single-qubit Clifford gates used to change the Pauli basis. We now turn to examining the model error of our noise estimates.

5.4 ACES model violation

Now that we are implementing ACES experimentally, rather than in simulation, we do not have ground truth values against which to compare our noise estimates. Instead, we

examine how well the data are fit by our noise model. In this section, we describe how this can be done with standard linear regression theory by computing the generalised residual sum of squares. The generalised residual sum of squares is expected to obey a chi-squared distribution if our noise model is correct, that is, if the data are consistent with a time-independent circuit-level Pauli noise model. Thus the z-score of the generalised residual sum of squares, the number of standard deviations from the mean, is a model violation score that assesses model error and the extent to which the assumptions of the noise model are violated.

Recall from Section 2.3, and particularly Section 2.3.4, that ACES noise estimates are obtained by solving the linear regression problem

$$\boldsymbol{b} = A\boldsymbol{x} + \boldsymbol{\varepsilon}. \tag{5.2}$$

The circuit log-eigenvalues $\boldsymbol{b}$ are related to the gate log-eigenvalues $\boldsymbol{x}$ by the design matrix A, and the error variables are distributed according to a multivariate normal distribution $\boldsymbol{\varepsilon} \sim \mathcal{N}_M(0, \Omega')$, where Ω' is the circuit log-eigenvalue estimator covariance matrix. Recalling the generalised least squares discussion in Section 2.B, notice that we can decorrelate and standardise the variances with a linear transformation. Specifically, consider the Cholesky factorisation $\Omega' = LL^\intercal$, and then left-multiply Equation 5.2 by L^{-1} to obtain

$$L^{-1}\boldsymbol{b} = L^{-1}A\boldsymbol{x} + L^{-1}\boldsymbol{\varepsilon}, \tag{5.3}$$

where now $L^{-1}\boldsymbol{\varepsilon} \sim \mathcal{N}_M(0, I_M)$.

Suppose we have an estimate of the circuit log-eigenvalues $\boldsymbol{b}$. Then referencing Equation 5.3, we see that the generalised least squares estimator for the gate-log-eigenvalues $\hat{\boldsymbol{x}}$ is given

$$\hat{\boldsymbol{x}} = \left(A^\intercal \Omega'^{-1} A\right)^{-1} A^\intercal \Omega'^{-1} \boldsymbol{b}. \tag{5.4}$$

Then the *residuals* are

$$\hat{\boldsymbol{r}} = \left(\boldsymbol{b} - A\hat{\boldsymbol{x}}\right) = \left(I - A\left(A^\intercal \Omega'^{-1} A\right)^{-1} A^\intercal \Omega'^{-1}\right)\boldsymbol{b} = (I - H)\boldsymbol{b}, \tag{5.5}$$

where we call H the *hat matrix* because

$$\hat{\boldsymbol{b}} = A\hat{\boldsymbol{x}} = H\boldsymbol{b}. \tag{5.6}$$

The hat matrix H is a projection matrix, though in our case it is not orthogonal, and hence so too is $I - H$. We can also define the *generalised residuals* as

$$\hat{\boldsymbol{h}} = L^{-1}\hat{\boldsymbol{r}} = L^{-1}(I - H)\boldsymbol{b}, \tag{5.7}$$

and the *generalised residual sum of squares* as

$$\hat{\boldsymbol{g}}^\intercal \hat{\boldsymbol{g}} = \hat{\boldsymbol{r}}^\intercal \Omega'^{-1} \hat{\boldsymbol{r}} = \left(\boldsymbol{b} - A\hat{\boldsymbol{x}}\right)^\intercal \Omega'^{-1} \left(\boldsymbol{b} - A\hat{\boldsymbol{x}}\right) = \boldsymbol{b}^\intercal \left(\Omega'^{-1}(I - H)\right)\boldsymbol{b}. \tag{5.8}$$

Indeed this is the squared Mahalanobis distance, introduced in Section 3.A, between $\boldsymbol{b}$ and $A\hat{\boldsymbol{x}}$.

Standard linear regression theory tells us that the generalised residual sum of squares in Equation 5.8 obeys a chi-squared distribution with $K = M - N$ degrees of freedom, where M is the number of circuit eigenvalues and N is the number of gate eigenvalues, that is, $\hat{\boldsymbol{g}}^\intercal \hat{\boldsymbol{g}} \sim \chi^2_K$. To see that this is true, notice that the Ω'^{-1} term in Equation 5.8 serves to

Table 5.1: Model violation z_χ, given in Equation 5.9, of ACES ordinary and combined noise estimates, as well as the projected noise estimates, for the syndrome extraction circuit in Figure 5.2 on IBM Fez. Model violations are shown for a range of shots per randomisation S'_R, and roughly halve for each factor of two reduction in S'_R.

S'_R	Ordinary z_χ	Combined z_χ	Proj. ordinary z_χ	Proj. combined z_χ
64	247.8	315.7	1890	2121
32	127.6	161.8	948.0	1066
16	62.96	81.02	472.8	533.4
8	32.51	40.89	236.8	266.1

decorrelate and standardise the variances, whereas the projection matrix $I - H$ has trace $M - N$ since both A and Ω' have full rank. Thus we are left with the sum of $K = M - N$ squared standard normal variables, which is exactly a chi-squared distribution with K degrees of freedom. This distribution has mean K and standard deviation $\sqrt{2K}$, so we define the *model violation* z_χ as the z-score of the residual sum of squares

$$z_\chi = \frac{\hat{\boldsymbol{g}}^\intercal \hat{\boldsymbol{g}} - K}{\sqrt{2K}}, \tag{5.9}$$

a more principled but essentially equivalent version of the reduced chi-squared statistic $\left(\hat{\boldsymbol{g}}^\intercal \hat{\boldsymbol{g}}\right)/K$. We validate that z_χ approximately follows a standard normal distribution in numerical simulations. Hence the model violation z_χ allows us to assess whether the observed circuit log-eigenvalues $\boldsymbol{b}$ are consistent with the time-independent circuit-level Pauli noise model entailed by Equation 5.2.

There are two important observations to make about the model violation. First, a large model violation indicates high confidence that the noise model is violated. It does not measure the extent to which the noise model must be modified to fit the data well. Second, note that Equation 5.8 features the unprojected estimates of the gate log-eigenvalues. These estimates need not correspond to valid probability distributions so as discussed previously, particularly in Section 3.A, we must project our estimates into the probability simplex. A version of this model violation calculated with these projected estimates does not even approximately follow a standard normal distribution in numerical simulations.

We are now ready to examine the model violation of our noise estimates obtained in Section 5.3. Table 5.1 shows the model violation z_χ of ACES ordinary and combined noise estimates, as well as the projected noise estimates, for the syndrome extraction circuit in Figure 5.2 on IBM Fez. The model violations are shown for a range of shots per randomisation S'_R, from the full 64 collected from the device down to 8. Below $S'_R = 8$, for which the shot budget S' is reduced from the original 5×10^6 to $S' = 6.25 \times 10^5$, too few shots are collected to ensure the that the estimate of the circuit log-eigenvalue estimator covariance matrix is positive-definite.

We observe in Table 5.1 that model violations roughly halve for each factor of two reduction in S'_R. This demonstrates that reducing the number of shots per randomisation is valuable even below $S'_R = 64$ down to $S'_R = 8$. If we suppose this trend continues down to full randomisation with $S'_R = 1$, this extrapolation indicates that our ACES noise

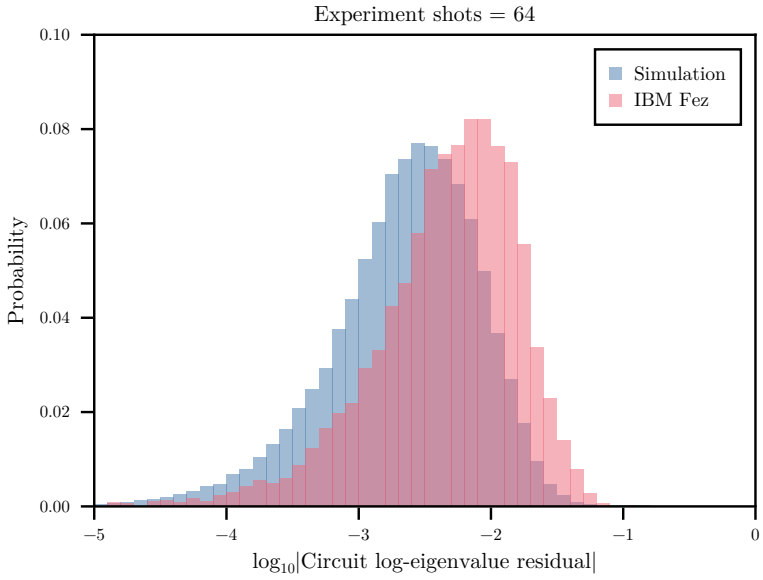

Figure 5.3: Histogram of the circuit log-eigenvalue residuals with $S'_R = 64$ shots per randomisation, comparing experimental data from IBM Fez (red) to simulated data (blue). The simulations are for 20 trials of ACES with the same shot budget where gate noise values were given by the device noise estimates.

estimates in this context may be able to achieve model violations of around 4 for the ordinary noise model and 5 for the combined noise model.

As a point of comparison, we turn to gate set tomography (GST) [62], a rigorous but expensive noise characterisation protocol. GST experiments commonly consider what is essentially this exact model violation quantity, though in that context it is derived from more involved statistical theory. In experimental implementations of GST, such as [149], a model violation of under 5 is a benchmark for a low model violation. This suggests that with full randomisation, noise in IBM Fez is well described by a time-independent circuit-level Pauli noise model.

It is also possible to perform model selection using the generalised residual sum of squares to calculate quantities such as the Akaike information criterion (AIC) or the Bayesian information criterion (BIC) [109]. These quantities formally prefer the ordinary noise model. However, as discussed previously, the SPAM noise estimates for the ordinary noise model differ across Pauli bases more than can reasonably be expected, given the small number of single-qubit Clifford gates used to change the Pauli basis. Therefore the combined noise model is more parsimonious, and so we do not consider this model selection procedure in further detail here.

We can also directly compare the residuals, as defined in Equation 5.5, for the device data against simulated residuals generated from the device noise estimates we obtained. For the combined noise model, we show the residual histograms at $S'_R = 64$ in Figure 5.3 and at $S'_R = 8$ in Figure 5.4. Note that we display the logarithm of the absolute value of the residuals, which is equivalent to the logarithm of the squared residuals up to a factor of two. As suggested by their relative model violations, reducing the number of shots per randomisation makes the device residuals more consistent with the simulated residuals. In particular, we notice that the device residual distribution is shifted to the right of the

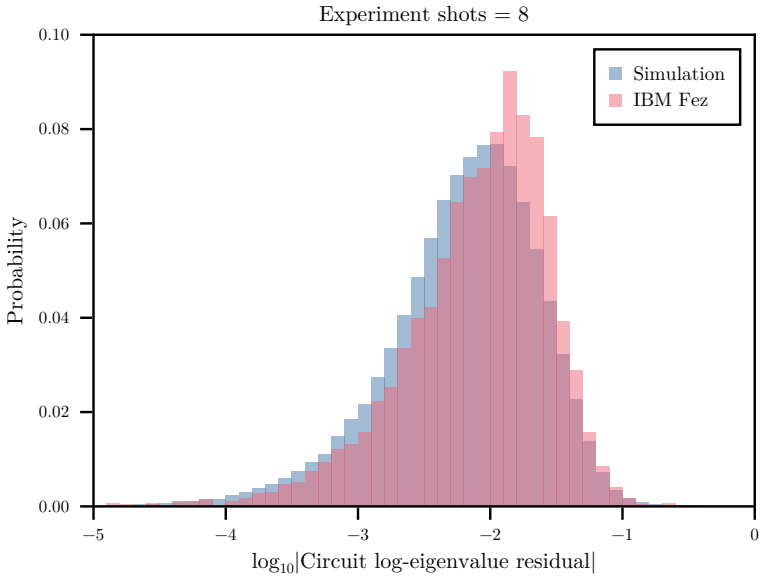

Figure 5.4: Histogram of the circuit log-eigenvalue residuals with $S'_R = 8$ shots per randomisation, comparing experimental data from IBM Fez (red) to simulated data (blue). The simulations are for 20 trials of ACES with the same shot budget where gate noise values were given by the device noise estimates.

simulated residual distribution—which corresponds to larger residuals—but this shift is reduced as S'_R is decreased from 64 to 8. Nevertheless, there is still a noticeable shift at $S'_R = 8$, suggesting that further gains could be made by further reductions to the number of shots per randomisation.

Our results indicate that the time-independent circuit-level Pauli noise model estimated by ACES is a good model for noise in IBM Fez. With full randomisation, which could be performed with the hardware-efficient randomisation methods of [101], we believe that ACES can attain model violations similar to those found in GST. GST similarly assumes time-independent noise but is otherwise a highly rigorous noise characterisation technique, although it struggles to scale beyond a few qubits. This indication that the circuit-level Pauli noise model can achieve very low model errors is a very encouraging sign for the utility of scalable Pauli noise characterisation techniques such as ACES.

5.5 Heavy hexagon memory

Finally, we are ready to operate IBM Fez as a heavy hexagon memory. We use the syndrome extraction circuit given in Figure 5.2 to construct memory experiment circuits, with two major variants. When implementing ACES on IBM Fez, our Qiskit circuits included *barriers* between the layers to ensure we only ever perform the appropriate layers of gates, and dynamical decoupling on the idling qubits during measurements. In one variant, we do the same for memory experiments. In the other, we do not include barriers except around measurements and perform dynamical decoupling throughout the circuit, as the barriers prevent the Qiskit tranpiler from optimising the circuit.

We present preliminary results for a memory experiment in the $|0\rangle$ basis in Figure 5.5. The memory experiment was decoded with PyMatching [45], a fast correlated matching

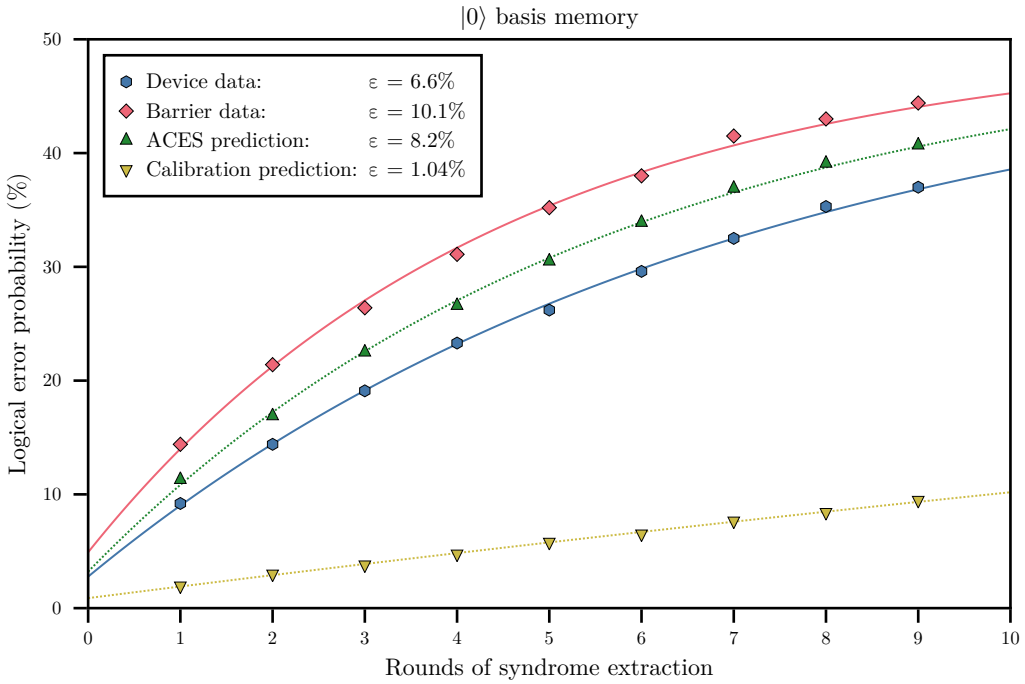

Figure 5.5: Logical error probability as a function of the number of rounds of syndrome extraction for a heavy hexagon memory in the $|0\rangle$ basis on IBM Fez, for an ordinary memory circuit and a memory circuit with barriers, decoded with a MWPM decoder informed by device calibration data. Predicted logical error probabilities derived from ACES noise estimates and publicly available device calibration data are also shown. The logical error per round ε is obtained by fitting the logical error probability as an exponential decay in the number of rounds. Whereas the ACES prediction lies roughly halfway between the memory experiments with and without barriers, the device calibration data predicts vastly less noise than is observed in the memory experiments.

decoder, informed by device calibration data, and collected 4096 shots for a number of rounds ranging from 1 to 9. The logical error per round ε was obtained by fitting the logical error probability as an exponential decay in the number of rounds. Figure 5.5 also shows the memory experiment performance predicted by ACES noise estimates and publicly available device calibration data.

We see that the memory experiment without barriers performs substantially better than the memory experiment with barriers, with the ACES prediction roughly halfway in between. By contrast, the device calibration data vastly overestimates device performance, predicting a logical error per round more than six times lower than the actual logical error per round. This demonstrates that the ACES noise estimates are a substantial improvement over publicly available device calibration data.

Although the ACES noise estimates outperform device calibration data, they do not match either memory circuit. Ideally, all three of these would coincide, but two major differences may explain the discrepancy. First, the memory circuits do not feature Pauli frame randomisation, unlike the circuits for ACES noise estimation. Second, the ordinary memory circuit is transpiled by the Qiskit transpiler. In the future, we plan to develop an optimised version of the syndrome extraction circuit expressed in terms

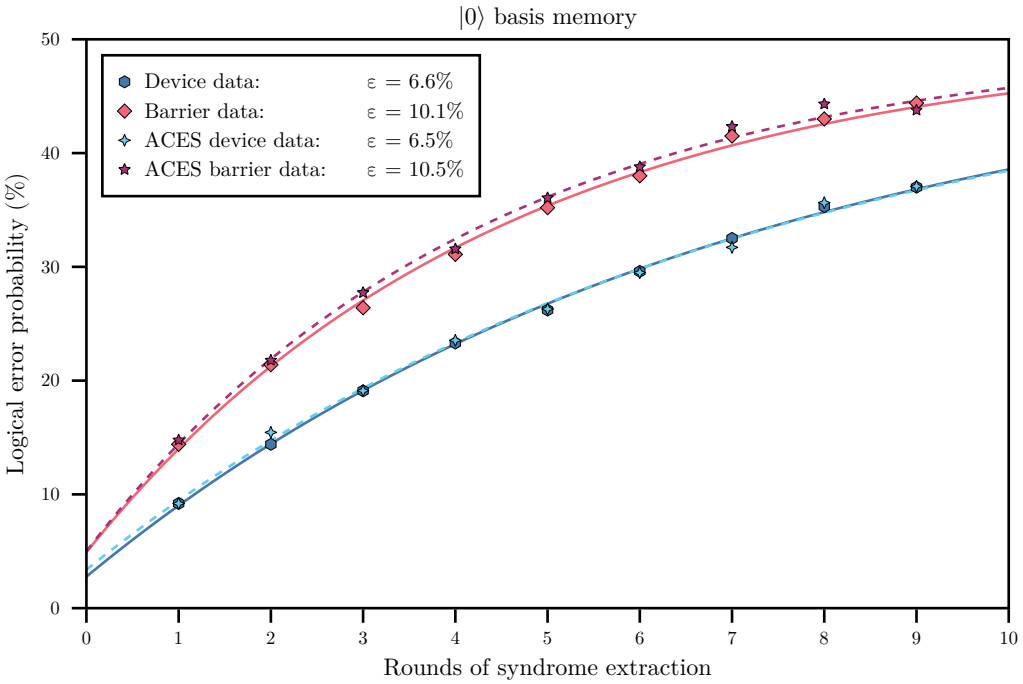

Figure 5.6: Logical error probability as a function of the number of rounds of syndrome extraction for a heavy hexagon memory in the $|0\rangle$ basis on IBM Fez, for an ordinary memory circuit and a memory circuit with barriers, decoded with a MWPM decoder informed by device calibration data. Logical error probabilities are also shown for the same memory experiments when the decoder is instead informed by ACES noise estimates. The logical error per round ε is obtained by fitting the logical error probability as an exponential decay in the number of rounds. Results are very similar regardless of the decoder prior.

of hardware-native gates, which here are controlled-Z gates as opposed to controlled-X gates. This should reduce or eliminate the need for circuit transpiling, and enable the insertion of barriers to ensure that the gate layers in the circuit are the same as in ACES noise characterisation experiments.

Next, we present preliminary results for noise-aware decoding of the same memory experiment in Figure 5.6. This compares logical error rates when the decoder is informed by device calibration data and ACES noise estimates. We see that the decoder prior does not have a substantial effect on logical error rates for both the ordinary memory experiment and the one with barriers.

There are several reasons that may explain why we do not see a substantial difference in the logical error rates between the different decoder priors in this setting. In particular, noise-aware decoding is not expected to substantially improve logical error rates for small codes with distance $d = 3$, as we saw in Chapter 3. We plan to investigate larger distance-5 codes, which are more likely to benefit from noise-aware decoding, in future work. Also, while the device calibration data may be a poor absolute predictor of noise in the device, it may still be a good predictor of the relative amounts of noise associated with different operations, which would suffice for decoding. It would therefore be interesting to compare these preliminary results to decoding informed by a depolarising noise model.

5.6 Conclusions

In this chapter, we have experimentally implemented ACES on IBM superconducting quantum devices and presented preliminary results for a heavy hexagon memory. Our findings suggest that the time-independent circuit-level Pauli noise model estimated by ACES is a good model for noise in IBM Fez and can achieve very low model errors. This is a very encouraging sign for the utility of scalable Pauli noise characterisation techniques such as ACES, as it suggests they can describe the essential features of quantum noise while remaining highly scalable. These findings also highlight the importance of full randomisation in Pauli noise characterisation techniques that rely on Pauli frame randomisation or randomised compiling enabled, for example, by [101]. We observe consistent improvements in the model violation when reducing the number of shots per randomisation from 64 down to 8, and believe that these improvements should continue down to full randomisation.

Our preliminary results for a heavy hexagon memory demonstrate that ACES noise estimates represent a substantial improvement over device calibration data in terms of performance prediction. However, we do not as yet see improvements in logical error rates when informing the decoder with ACES noise estimates as opposed to device calibration data. In the future, we plan to conduct more extensive tests of noise-aware decoding, including examining a larger distance-5 code, trying different decoders such as BeliefMatching [83], and comparing our results to decoding informed with a depolarising noise model.

We also plan to make several improvements to reduce the model error of our noise estimates. Including repetitions of the syndrome extraction circuit without measurement in ACES noise characterisation experiments may reduce model error and will enable direct testing of its model violation. Expressing the syndrome extraction circuit in terms of hardware-native gates should mitigate the need to transpile the syndrome extraction circuit to maximise performance, aligning memory circuits with those we characterise with ACES. We would also like to improve how ACES characterises mid-circuit measurement noise, following recent work [111, 112].

In the future, it would be interesting to search over the gauge freedom in the Pauli noise model following [96], as is often done in GST [62]. This may allow us to find a gauge that minimises the model violation increase associated with projecting noise estimates into the probability simplex, as our noise model implicitly fixes the gauge by fully associating SPAM noise with measurements. Device hardware improvements targeting improved measurements, the noisiest operation on IBM Heron devices, would also close the co-design loop opened in our work here.

Acknowledgements

This work was supported by the Australian Research Council Centre of Excellence for Engineered Quantum Systems (CE170100009) and the U.S. Army Research Office (W911NF-23-S-0004).

5.A ACES noise estimates

This appendix presents the full noise characterisation results obtained from IBM Fez. We present noise estimates for the combined noise model, which treats Pauli X, Z, and Y basis SPAM noise as a single parameter for each qubit, noting that we associate SPAM noise with measurements, implicitly fixing the gauge of the noise model. Table 5.2 shows the experimental design used to obtain these results, which features 50 tuples in its tuple set divided between 28 repeated tuples corresponding to deep circuits and 22 tuples corresponding to shallow circuits. Its circuit eigenvalues are estimated by 447 experiments before randomisations, and we collected 5×10^6 shots with 64 shots per randomised experiment and a minimum of 128 randomisations, which is reflected in the supplied shot weights. The circuit was placed on IBM Fez with the top left qubit at qubit 86 on the device.

Then we present the estimated Pauli error probabilities marginalised over gate Pauli orbits, as described in Section 3.A, for the gates in each layer of the circuit shown in Figure 5.2, as well as for measurements. The layout of the histograms reflects the qubit layout and connectivity of IBM Fez. We also supply error bars calculated from the gate error probability estimator covariance matrix, which crucially do not account for model error in the noise estimates, and so underestimate the true error associated with these estimates. In particular, we make the most assumptions about measurement noise and expect model error to be most substantial for these estimates. We observe that measurement and reset, as well as qubit idling during measurement and reset, are the noisiest operations by a large margin. We also see that noise estimates for the same gates across different layers tend to be similar despite being estimated independently. This is an encouraging sign that ACES has accurately characterised noise in IBM Fez.

Table 5.2: The optimised experimental design for the syndrome extraction circuit of a distance-3 heavy hexagon code shown in Figure 5.2.

Shot weight	Tuple	Repetition number
0.014746	(1)	39
0.014746	(2)	39
0.014746	(3)	39
0.014746	(4)	39
0.014746	(5)	39
0.014746	(6)	39
0.014746	(7)	39
0.014746	(8)	39
0.014746	(9)	39
0.014746	(10)	39
0.014746	(11)	39
0.014746	(12)	39
0.014746	(13)	39
0.014746	(14)	39
0.014746	(1)	79
0.014746	(2)	79
0.014746	(3)	79
0.014746	(4)	79
0.014746	(5)	79
0.014746	(6)	79
0.014746	(7)	79
0.014746	(8)	79
0.014746	(9)	79
0.014746	(10)	79
0.014746	(11)	79
0.014746	(12)	79
0.014746	(13)	79
0.014746	(14)	79
0.023847	(2, 6)	
0.050689	(3, 11)	
0.021313	(3, 13)	
0.038247	(4, 8)	
0.037787	(4, 13)	
0.050574	(5, 9)	
0.016935	(7, 14)	
0.041588	(10, 7)	
0.030529	(10, 14)	
0.014746	(12, 2)	
0.027533	(14, 9)	
0.018433	(14, 12)	
0.004915	(15, 15)	
0.014746	(5, 1, 5)	
0.014976	(7, 9, 11)	
0.014746	(7, 11, 5)	
0.033409	(8, 12, 2)	
0.068200	(13, 1, 6)	
0.004915	(15, 15, 15)	
0.014746	(3, 6, 13, 1)	
0.014746	(3, 4, 6, 8, 2, 1)	
0.029492	(6, 10, 4, 9, 8, 14, 12, 2)	

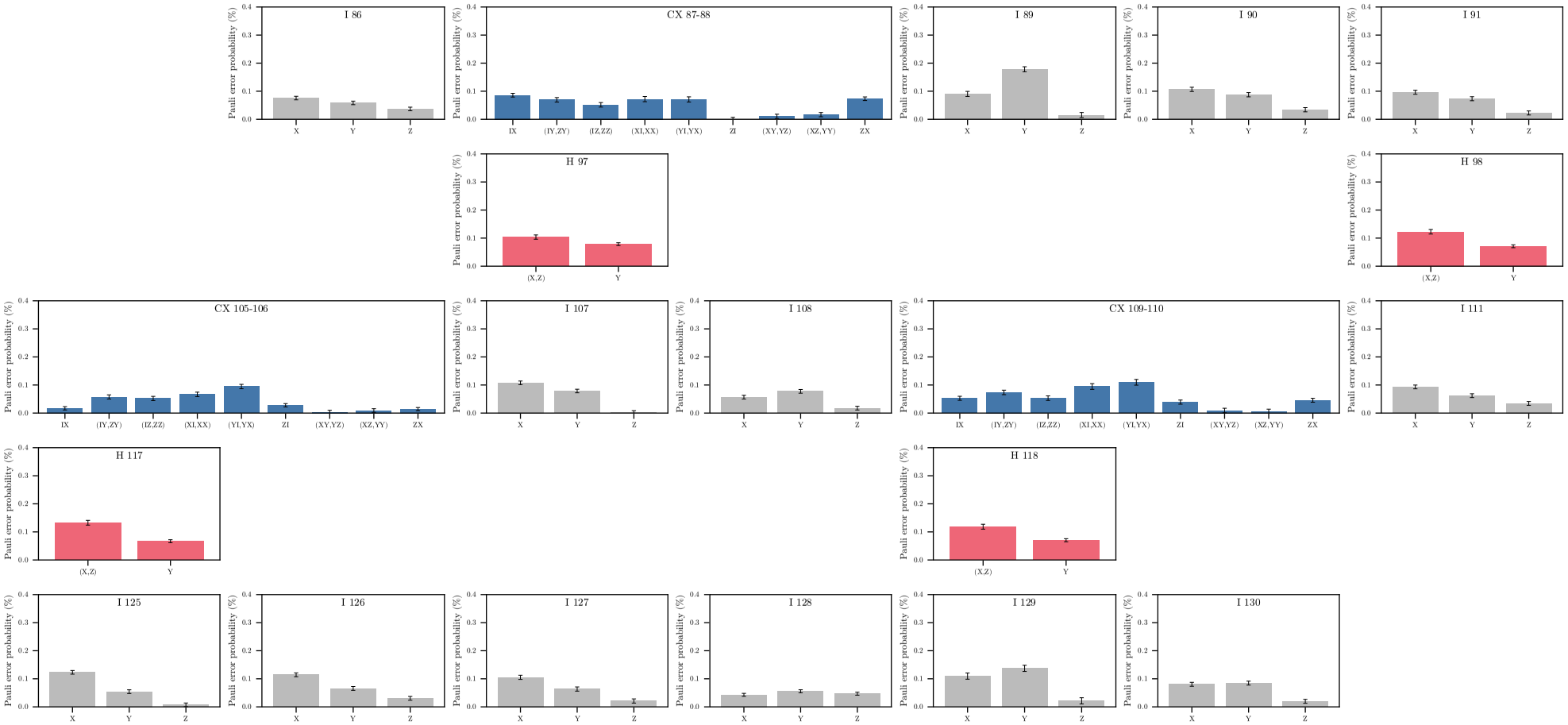

Figure 5.7: ACES orbit Pauli error probability estimates, with error bars indicating two standard deviations for visibility calculated from the noise model, for the gates in layer 1 of Figure 5.2 on IBM Fez.

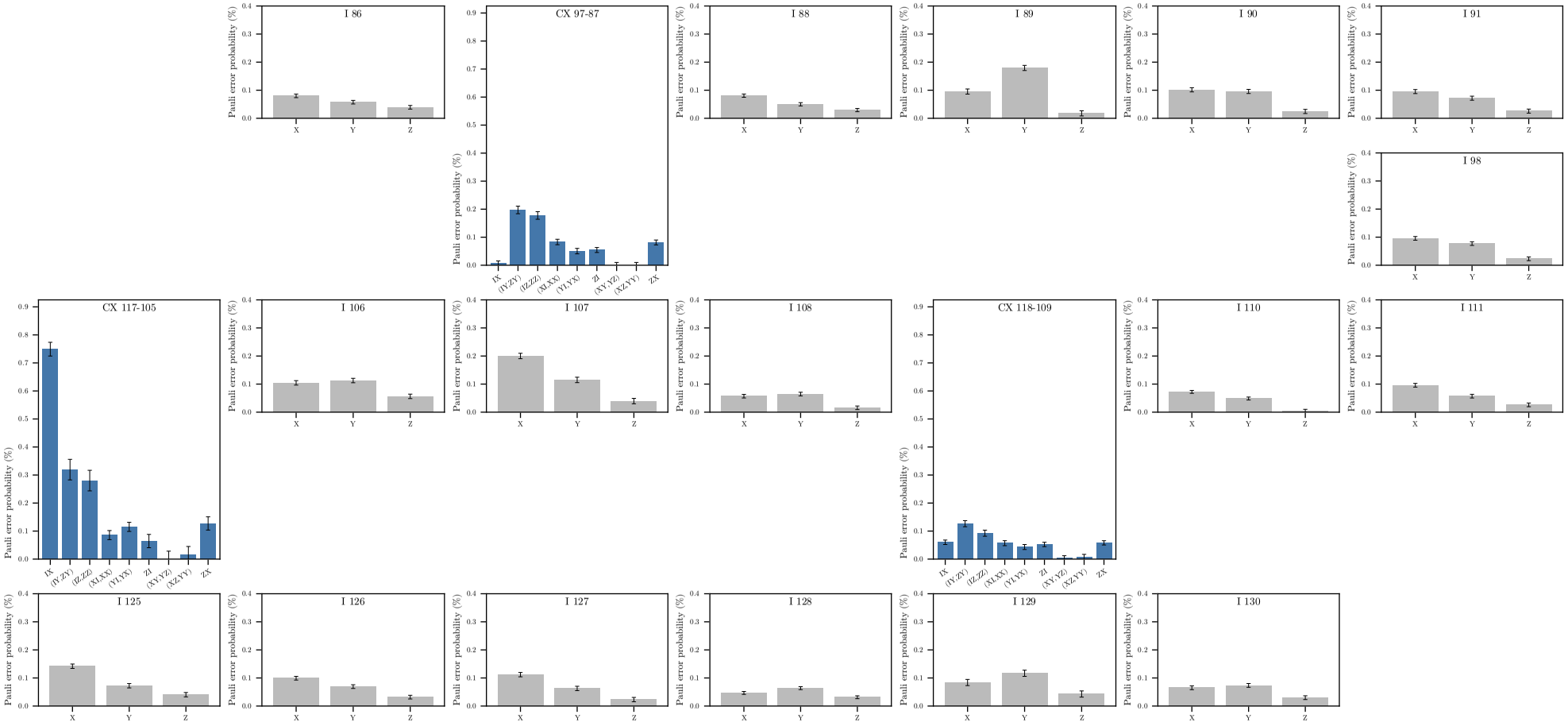

Figure 5.8: ACES orbit Pauli error probability estimates, with error bars indicating two standard deviations for visibility calculated from the noise model, for the gates in layer 2 of Figure 5.2 on IBM Fez.

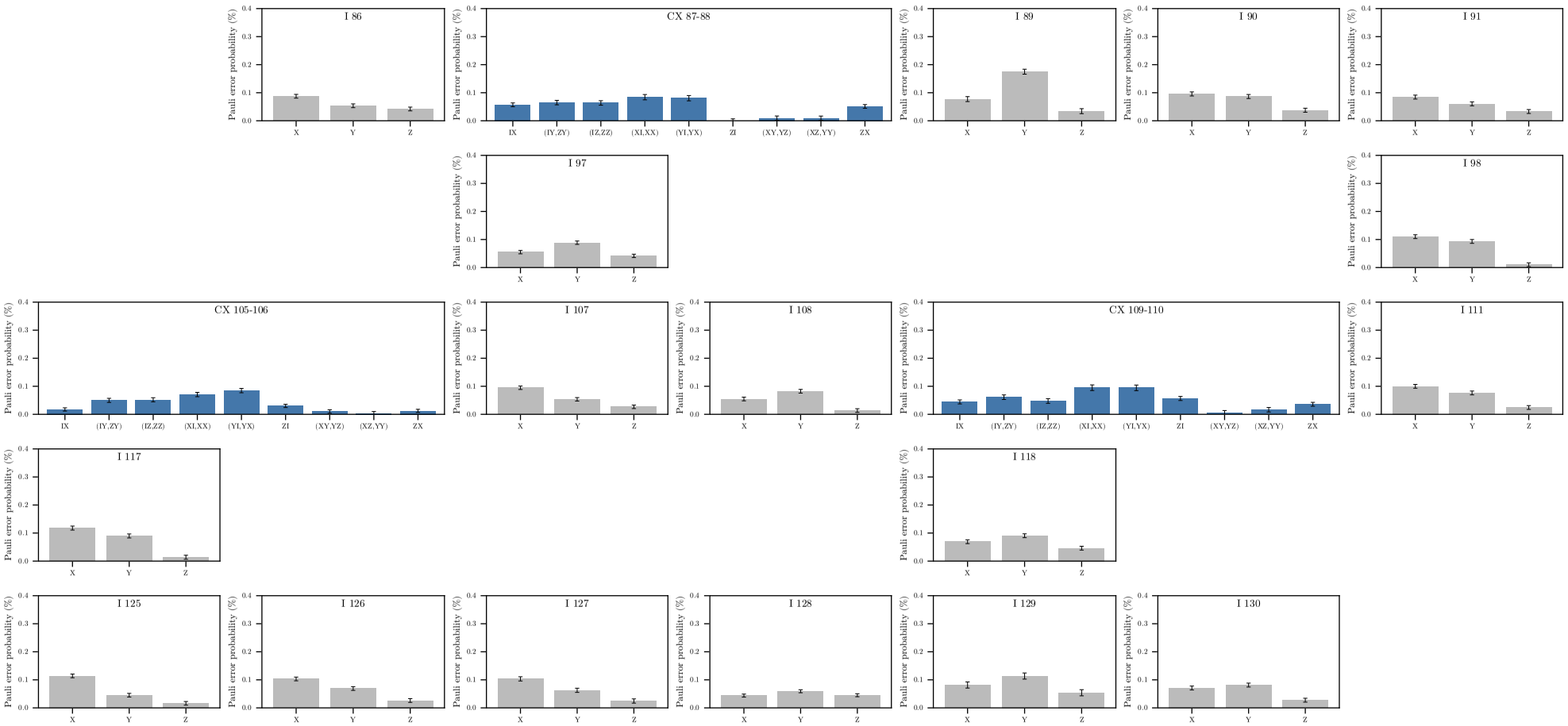

Figure 5.9: ACES orbit Pauli error probability estimates, with error bars indicating two standard deviations for visibility calculated from the noise model, for the gates in layer 3 of Figure 5.2 on IBM Fez.

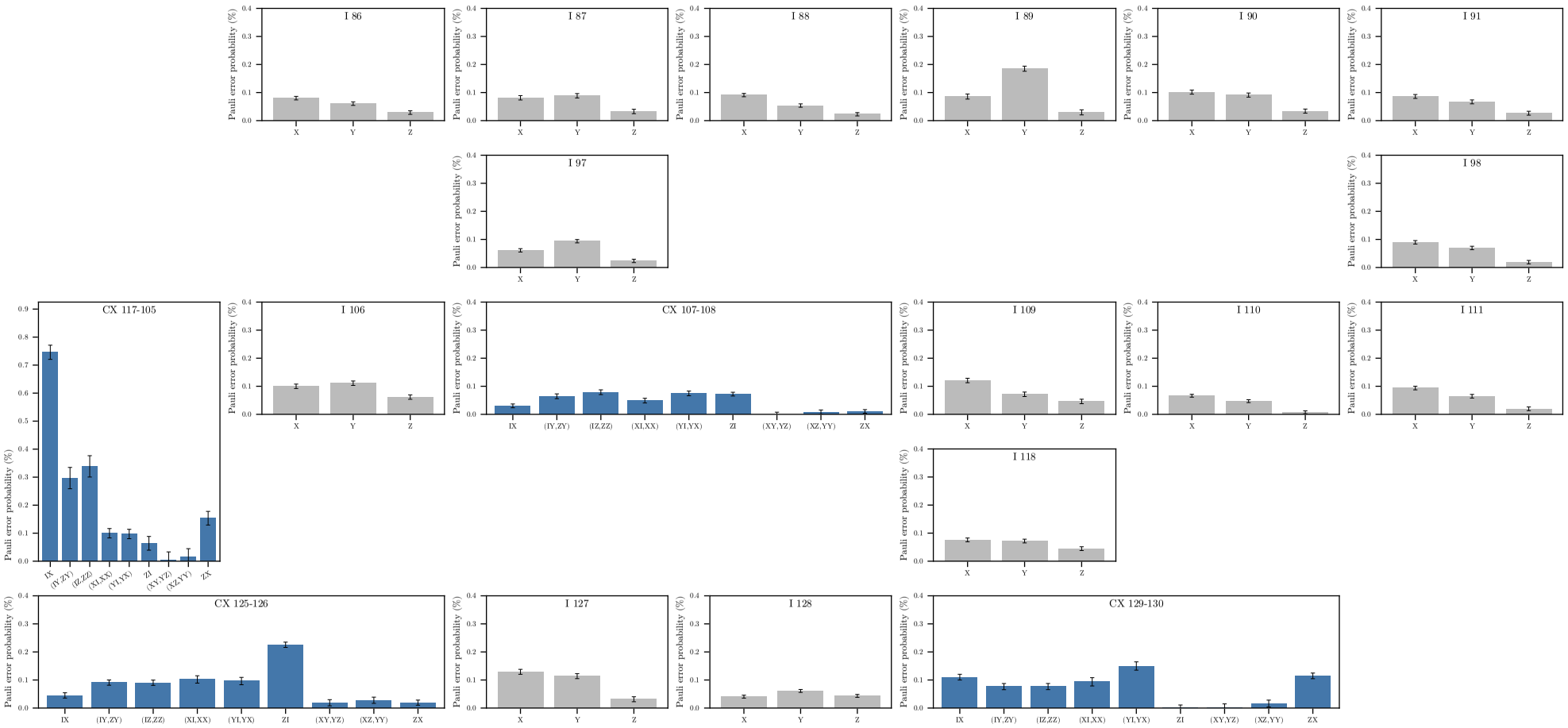

Figure 5.10: ACES orbit Pauli error probability estimates, with error bars indicating two standard deviations for visibility calculated from the noise model, for the gates in layer 4 of Figure 5.2 on IBM Fez.

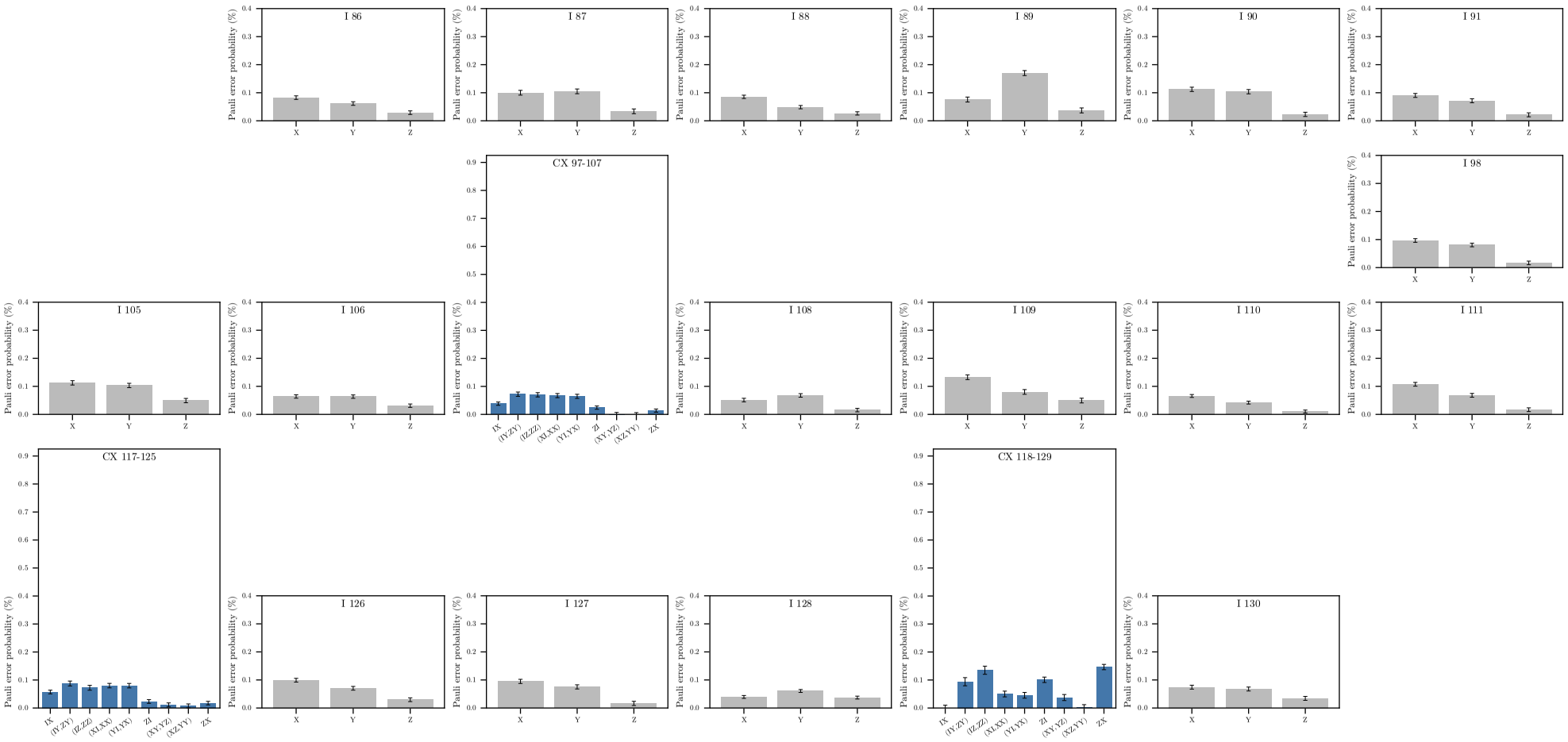

Figure 5.11: ACES orbit Pauli error probability estimates, with error bars indicating two standard deviations for visibility calculated from the noise model, for the gates in layer 5 of Figure 5.2 on IBM Fez.

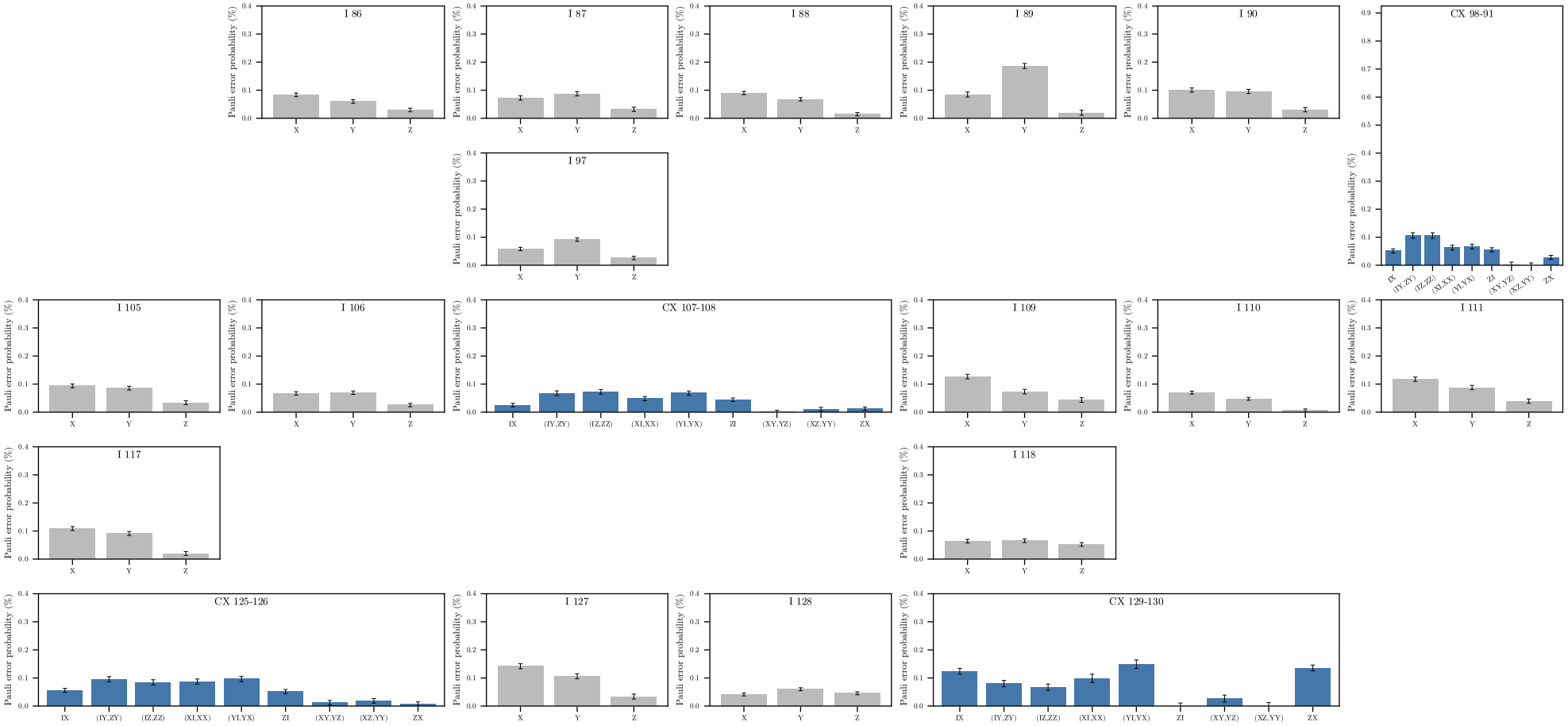

Figure 5.12: ACES orbit Pauli error probability estimates, with error bars indicating two standard deviations for visibility calculated from the noise model, for the gates in layer 6 of Figure 5.2 on IBM Fez.

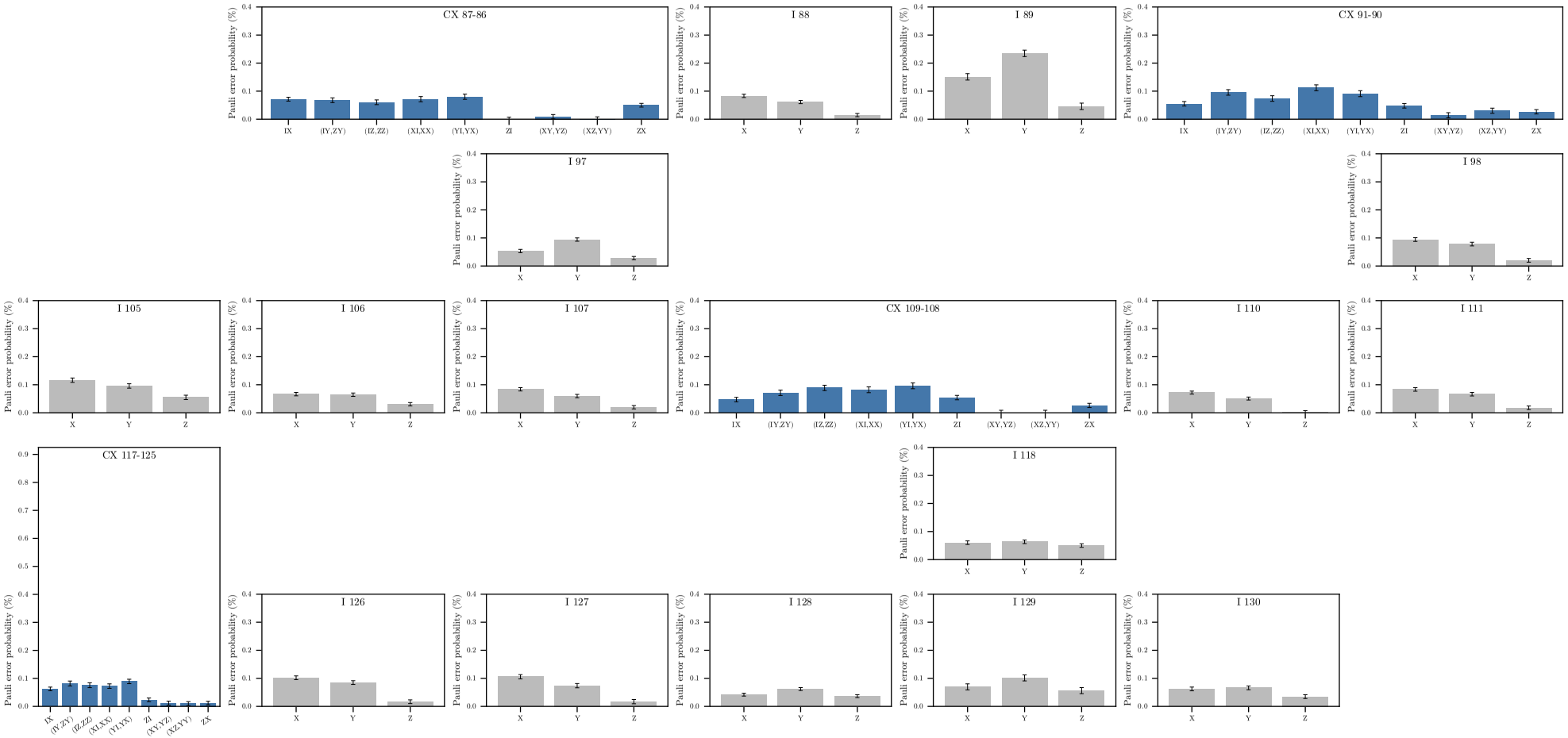

Figure 5.13: ACES orbit Pauli error probability estimates, with error bars indicating two standard deviations for visibility calculated from the noise model, for the gates in layer 7 of Figure 5.2 on IBM Fez.

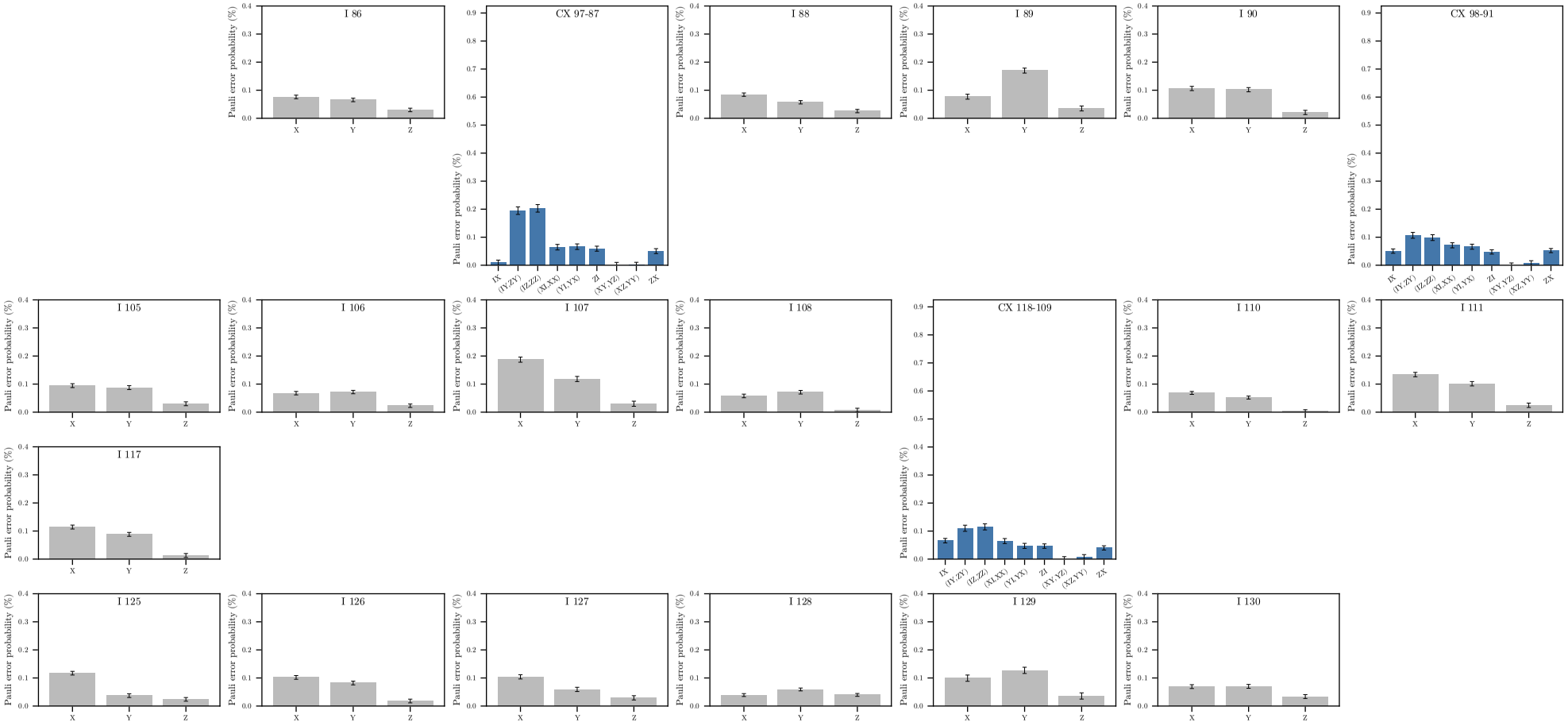

Figure 5.14: ACES orbit Pauli error probability estimates, with error bars indicating two standard deviations for visibility calculated from the noise model, for the gates in layer 8 of Figure 5.2 on IBM Fez.

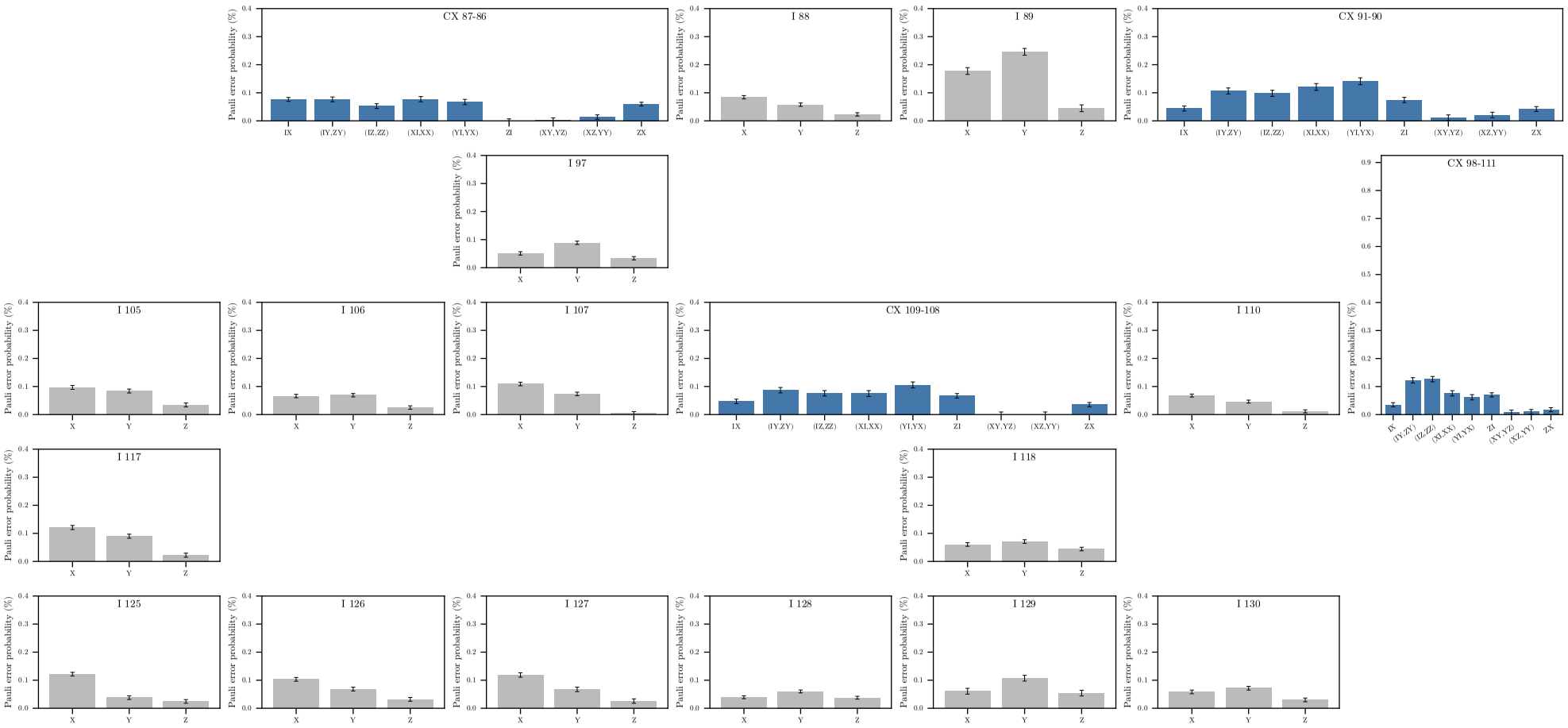

Figure 5.15: ACES orbit Pauli error probability estimates, with error bars indicating two standard deviations for visibility calculated from the noise model, for the gates in layer 9 of Figure 5.2 on IBM Fez.

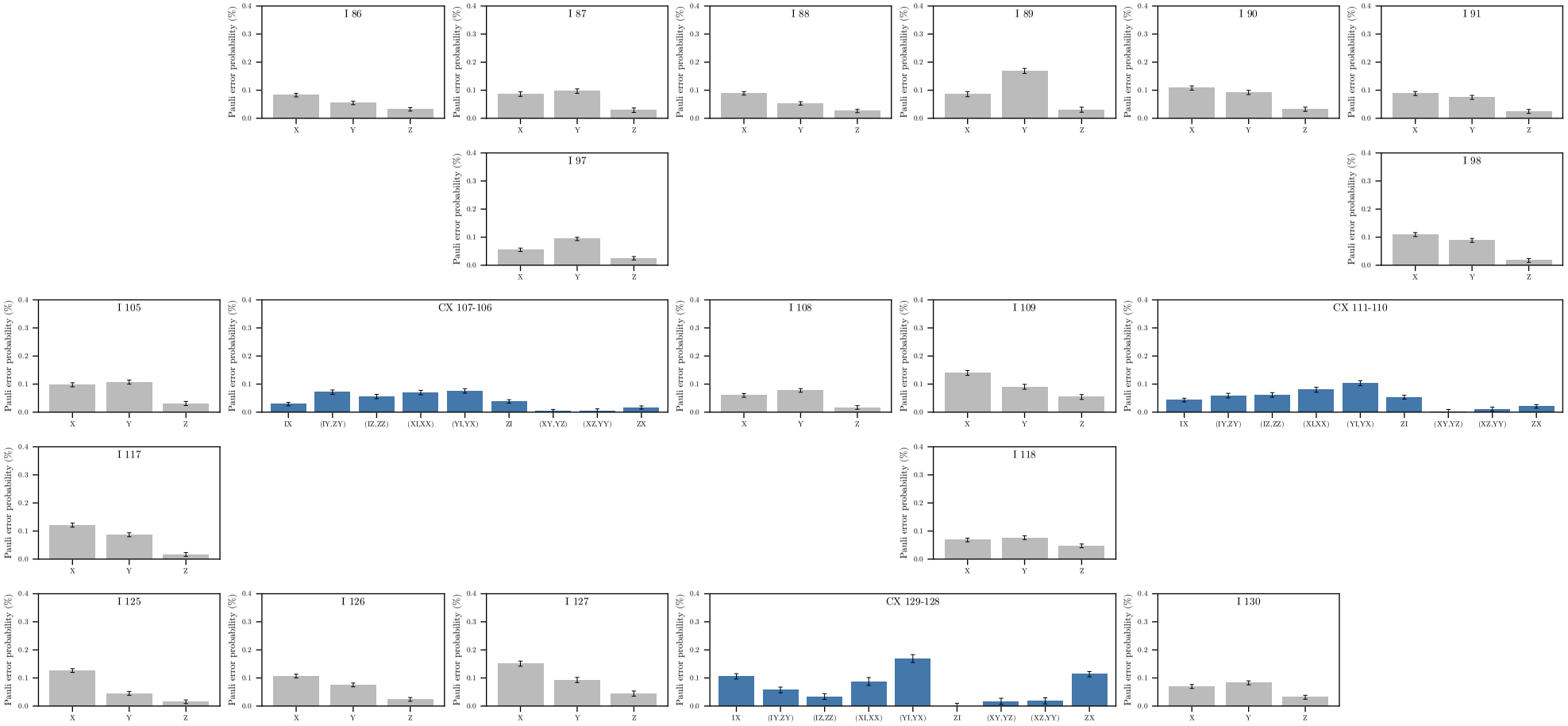

Figure 5.16: ACES orbit Pauli error probability estimates, with error bars indicating two standard deviations for visibility calculated from the noise model, for the gates in layer 10 of Figure 5.2 on IBM Fez.

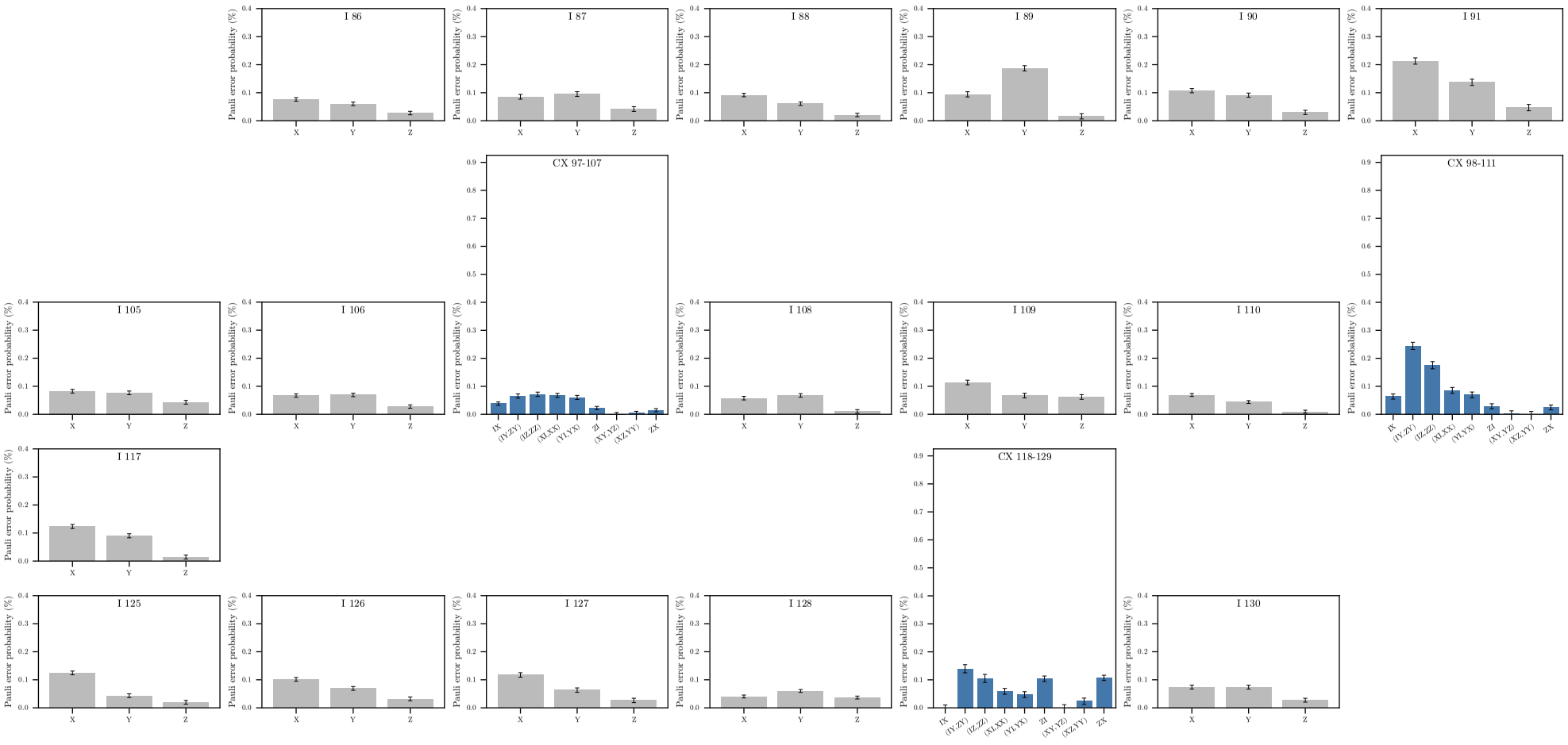

Figure 5.17: ACES orbit Pauli error probability estimates, with error bars indicating two standard deviations for visibility calculated from the noise model, for the gates in layer 11 of Figure 5.2 on IBM Fez.

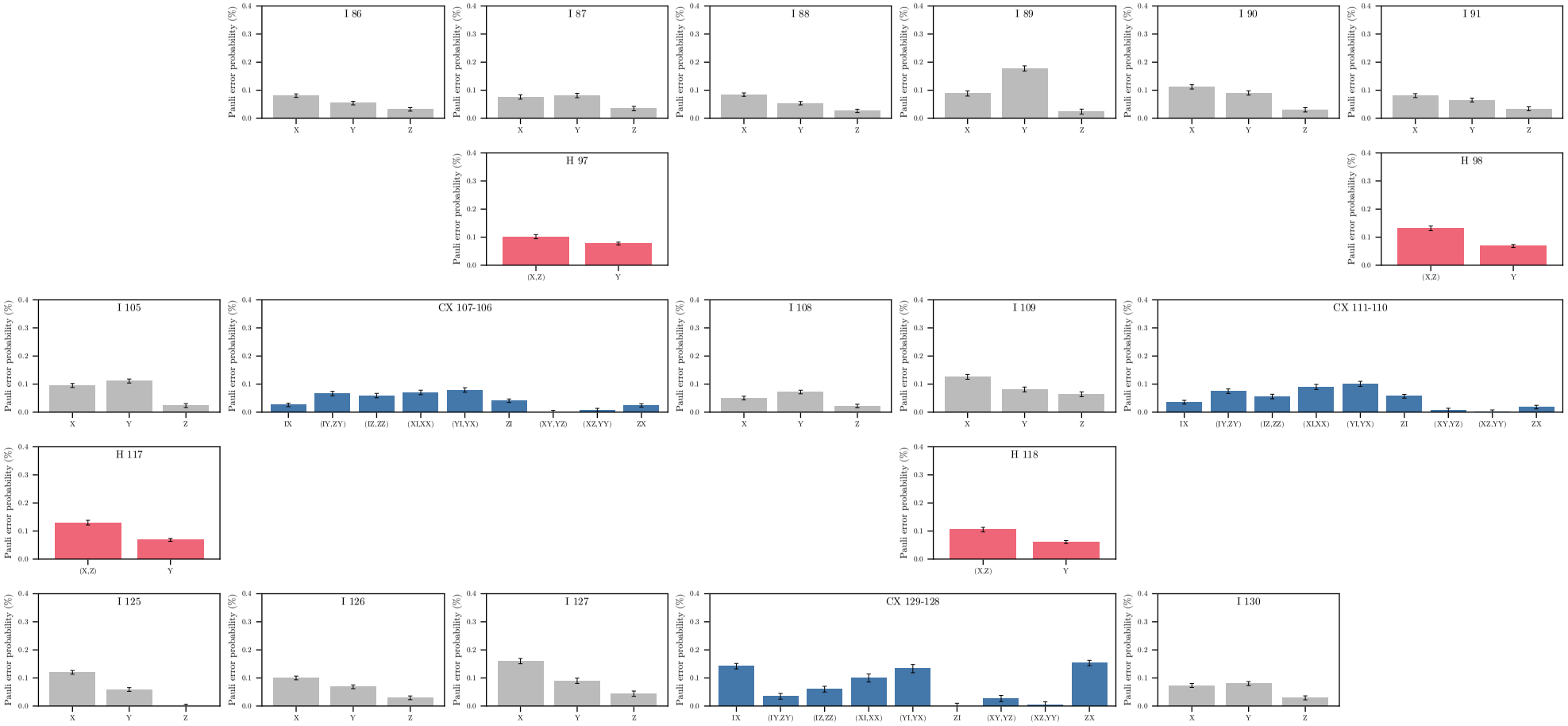

Figure 5.18: ACES orbit Pauli error probability estimates, with error bars indicating two standard deviations for visibility calculated from the noise model, for the gates in layer 12 of Figure 5.2 on IBM Fez.

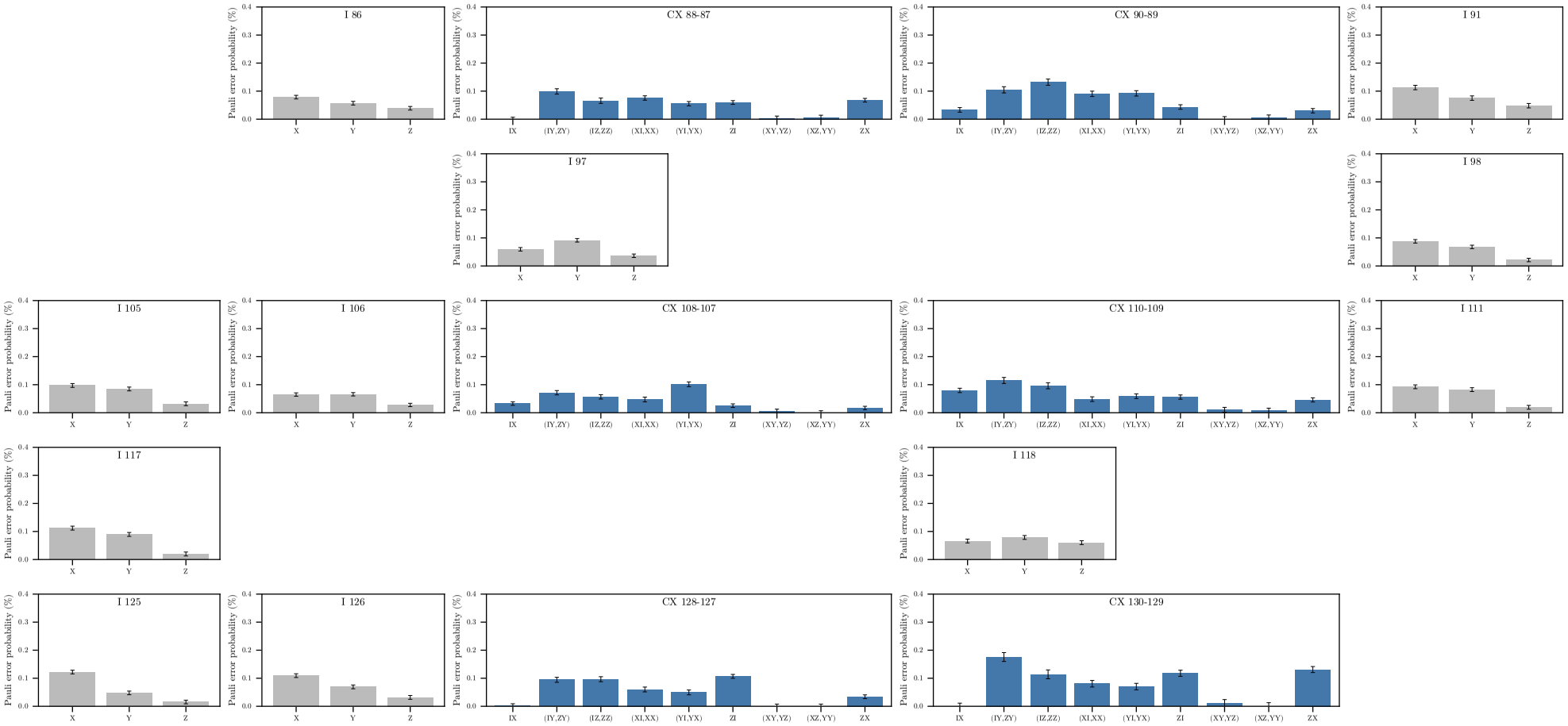

Figure 5.19: ACES orbit Pauli error probability estimates, with error bars indicating two standard deviations for visibility calculated from the noise model, for the gates in layer 13 of Figure 5.2 on IBM Fez.

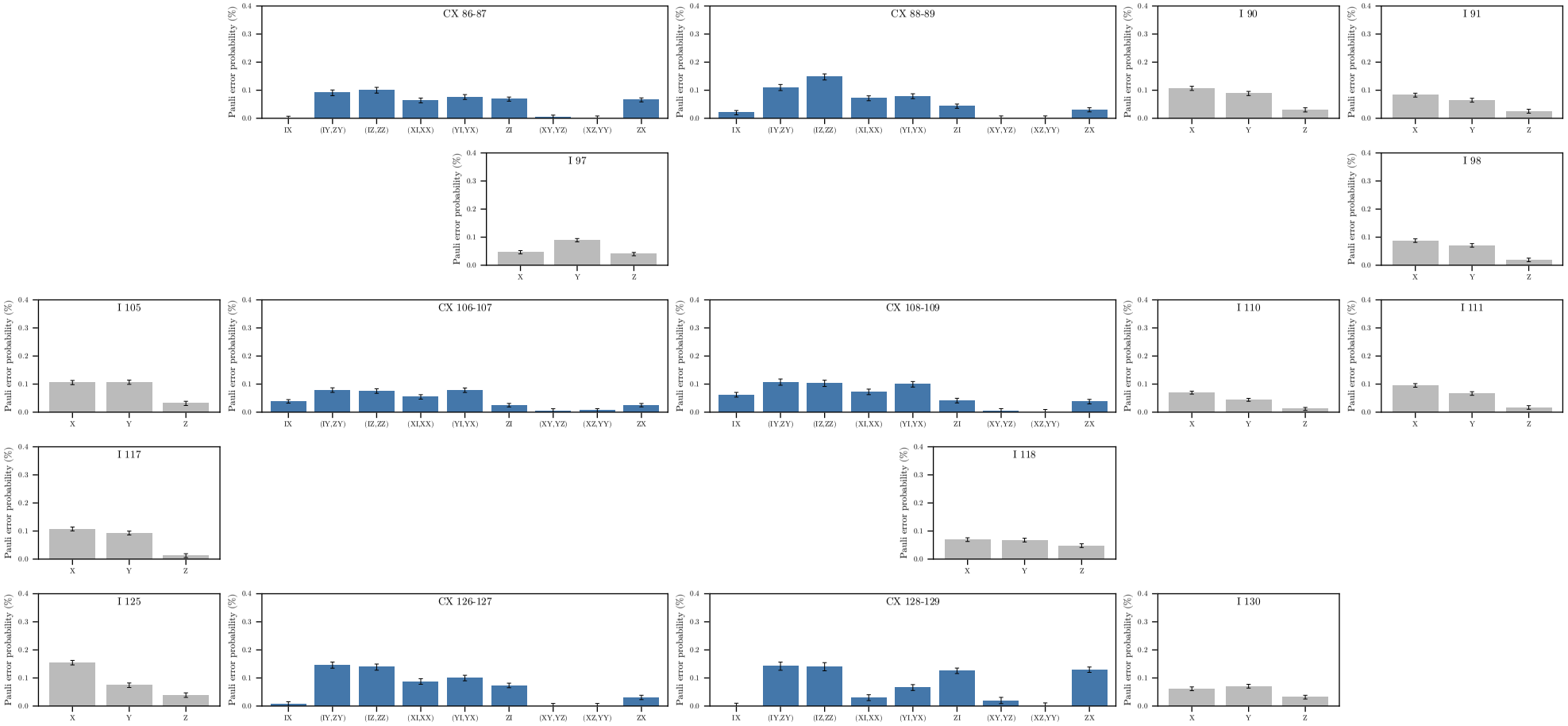

Figure 5.20: ACES orbit Pauli error probability estimates, with error bars indicating two standard deviations for visibility calculated from the noise model, for the gates in layer 14 of Figure 5.2 on IBM Fez.

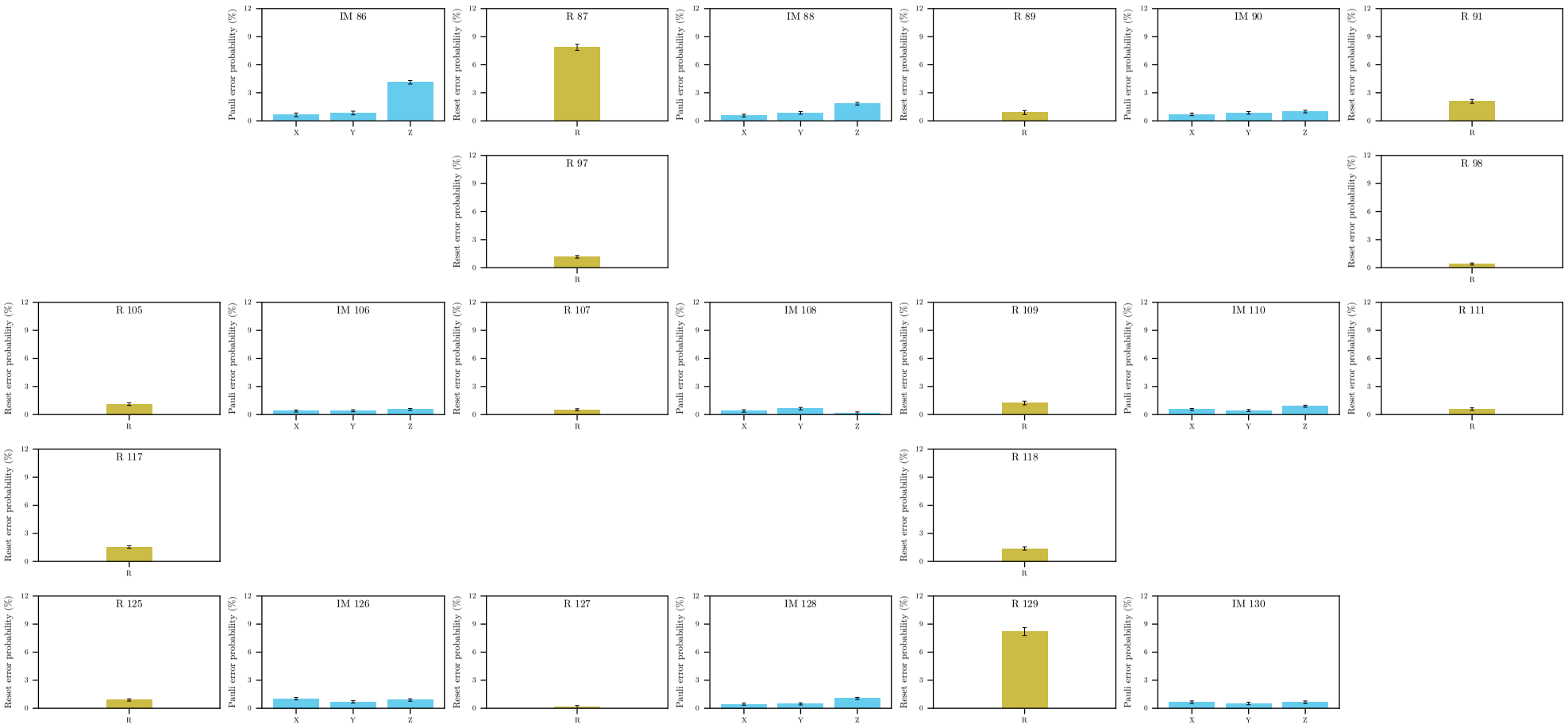

Figure 5.21: ACES reset error probability estimates, with error bars indicating two standard deviations for visibility calculated from the noise model, for the gates in layer 15 of Figure 5.2 on IBM Fez.

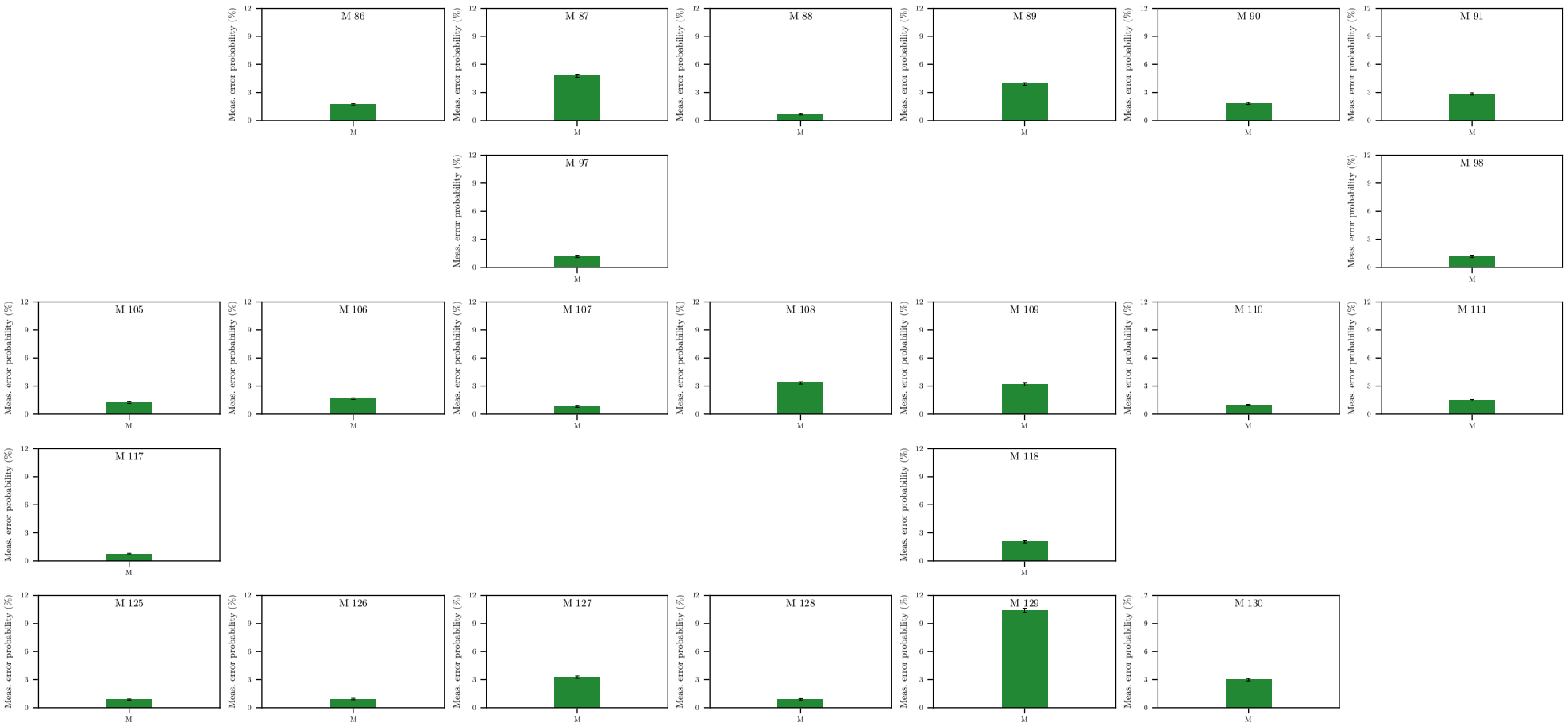

Figure 5.22: ACES measurement error probability estimates, with error bars indicating ten standard deviations for visibility calculated from the noise model, for Figure 5.2 on IBM Fez.

Chapter 6

Conclusion

> It all adds up to normality.
>
> — Greg Egan, *Quarantine*

In this thesis, we have introduced a scalable Pauli noise characterisation protocol suited to the context of quantum error correction and, in particular, syndrome extraction circuits. I also present the open-source Julia package QuantumACES, which provides a software implementation of this protocol. We have shown that the noise estimates can calibrate decoders for quantum error correcting codes, improving the error suppression factor of the code to exponentially reduce logical error rates. Lastly, we have presented ongoing work in experimentally implementing this protocol, where noise estimates were used to design an improved syndrome extraction circuit, predict circuit performance, and calibrate decoders. Our results indicate that the circuit-level Pauli noise model characterised by ACES can describe the essential features of quantum noise, and demonstrate the practicality and utility of Pauli noise characterisation in the context of fault-tolerant quantum computation. Pauli noise estimates can calibrate decoders to enable noise-aware decoding and inform the co-design of quantum error correcting codes, decoders, fault-tolerant circuits, and quantum devices.

6.1 Directions for future research

There are a number of directions for future work which I find particularly exciting. I will address the work presented in each chapter separately.

First, the scalable noise characterisation protocol in Chapter 2 uses a number of heuristic methods to optimise the design of noise characterisation experiments. These optimisation methods can certainly be improved, perhaps aided by deeper theoretical understanding of the performance characteristics of experimental designs, though it is unclear to me whether there is much room to improve performance. It would be interesting to examine whether optimised experimental designs can be transferred between the syndrome extraction circuits of quantum LDPC codes of different sizes, as we only considered topological codes. I expect this should be possible for quantum LDPC codes implemented on devices natively capable of the long-range interactions that enable efficient implementation of syndrome extraction circuits, that is, in an amount of time that does not grow with the code size. Also, the protocol relies on efficient classical

simulation of the circuit whose noise we aim to characterise; these circuits are not able to perform useful quantum computation. Nevertheless, it may be tractable to simulate fault-tolerant gadgets such as magic state distillation. If that is not possible, it will at least be possible to characterise a copy of the circuit where $T = \sqrt{S}$ is replaced with the efficiently simulable S, which may not substantially affect the noise in the circuit.

Then, in Chapter 3 we did not thoroughly investigate the performance benefits of calibrating decoder priors across a range of noise strengths and decoders, and doing so would be informative. For example, more accurate decoders may be able to better leverage noise information to obtain larger improvements in the logical error rate. It would be interesting to investigate how the performance depends on noise estimation accuracy, particularly the tradeoff between additive and relative precision estimates. It may be the case that decoders are relatively insensitive to noise parameters that can only be estimated to additive precision—perhaps with the exclusion of measurements—in which case the sample efficiency of calibrating decoder priors could be improved by biasing the noise estimation protocol towards relative precision estimation. A comparison of decoder performance when calibrated with a range of methods, giving each the same amount of time to sample from the quantum device, would also be interesting.

The Julia package QuantumACES, which I introduced in Chapter 4, currently only supports noise characterisation of mid-circuit resets. Adding support for noise characterisation of mid-circuit measurements, perhaps following [111, 112], so that they do not have to be treated as mid-circuit resets, would be particularly useful in the context of syndrome extraction circuits. Also, it may be useful to weight the reconstruction of the circuit eigenvalues of certain tuples more highly in the least squares estimation. For example, doing this for tuples that resemble the original circuit could reduce model error in the noise estimate. Optimising the choice of repeated tuples may also allow us to estimate more noise parameters to relative precision, analogous to germ selection in gate set tomography [62], and as recently discussed in the context of cycle benchmarking [150]. The benefits here may not be captured by the current relative precision figure of merit, so it may be necessary to derive an improved figure of merit to guide this optimisation.

Lastly, in Chapter 5 we implemented our noise characterisation and memory experiment in a small-scale code for which noise-aware decoding is not expected to produce much benefit. Also, we considered the heavy hexagon code, which lacks a threshold. We might find greater benefits in a code with a threshold such as the surface code. There are a number of modifications to our protocol that might reduce the model error of our noise estimates. Hardware-efficient randomised compiling would enable randomisation of the Pauli frame for each measurement shot. Including the syndrome extraction circuit in ACES noise characterisation experiments may reduce model error and will enable direct testing of its model violation. Expressing the syndrome extraction circuit in terms of hardware-native gates, here controlled-Z rather than controlled-X, should mitigate the need to transpile the syndrome extraction circuit to maximise performance, aligning memory circuits with those we characterise with ACES. Searching over the gauge freedom in the Pauli noise model, following [96], should allow us to find a gauge that minimises the model violation increase associated with projecting noise estimates into the probability simplex. Moreover, device hardware improvements targeting improved measurements would close the co-design loop initiated in our work here. Finally, it would also be interesting to use this noise characterisation protocol to perform probabilistic error cancellation.

6.2 Outlook

The field of quantum information is unlike many other scientific fields because of how closely it is linked to the great scientific and engineering challenge of building a useful fault-tolerant quantum computer. This challenge can be used to guide research. What role do I envisage, then, for scalable noise characterisation protocols in fault-tolerant quantum computers?

I expect that, as discussed previously, scalable noise characterisation protocols will inform the co-design of quantum error correcting codes, decoders, fault-tolerant circuits, and quantum devices. But this context demands less of a noise characterisation protocol than the more interesting context of their use in the operation of a fault-tolerant quantum computer, so I will focus on the latter.

Any noise characterisation protocol run on a fault-tolerant quantum computer must rapidly estimate noise to an accuracy sufficient for the intended applications of the noise estimates. If the protocol is too slow, frequently running it will substantially reduce the duty cycle of the quantum computer, rendering it impractical to deeply integrate the protocol into its operation. Therefore, any practical noise characterisation protocol must be fast to run, both in terms of quantum device time and classical processing overhead. The noise characterisation protocol presented in this thesis achieves this in the context of calibrating decoder priors for noise-aware decoding. For example, in superconducting architectures I expect this protocol will be able to achieve satisfactory accuracy in a few seconds of quantum device time and a few seconds of classical processing overhead.

How might we use this in a fault-tolerant quantum computer? Before starting to perform a computation, we would first use this noise characterisation protocol to check the calibration of the device and determine whether all of the operations are performing within desired bounds. Once this check passes, we would calibrate the decoder prior with these noise estimates and begin the computation. During the computation, the decoder prior would then be updated online from error syndrome data from syndrome data. If these updates indicate drift during the computation has become intolerable and predict that logical error rates are unacceptably high, we have two options. We can take the relevant portion of the device offline for recalibration, if the computer supports this through, for example, code concatenation. Otherwise, we may need to restart the computation entirely.

Large-scale quantum devices suitable for fault-tolerant quantum computation will need to mitigate crosstalk and drift. I believe that, as indicated by the results in this thesis, quantum error correction circuits, augmented with hardware-efficient randomisation, will closely adhere to a circuit-level Pauli noise model. This would obviate the need for more complicated and costly calibration methods that directly optimise the decoder prior according to the logical error rate. I hope, then, that the noise characterisation techniques presented here will prove to be a useful component of fault-tolerant quantum computers.

The architectures of future fault-tolerant quantum computers are radically underdetermined. The many competing hardware platforms will give rise to a large range of architectures. In classical computing, we have largely settled on the hardware platform of semiconductor transistors made from silicon. Yet we now see an increasing proliferation of a range of processor architectures specialised for different computational tasks, which is most evident in the ongoing build-out of data centres for artificial intelligence and machine learning. I expect that quantum computing will also see a range of architectures specialised for different computational tasks.

I see the intensifying and accelerating proliferation of computation as the core story of our time. Uninterrupted, it is sure to radically transform our civilisation. We do not see this future through a glass, darkly; it lies beyond a singularity through which we cannot see at all. Plans, then, are useless, but planning remains indispensable. Quantum computers will have a role in this story, though they may take the stage after its end is overdetermined. And as noise is a fundamental challenge for quantum computation, noise characterisation is reserved a role within the smaller story of fault-tolerant quantum computation. The results I have presented here indicate that noise characterisation is capable of rising to the challenge. I hope they contribute in some small way to this great story.

For in the end, there is always a way out.

— Hannu Rajaniemi, *The Causal Angel*

Bibliography

[1] M. A. Nielsen and I. L. Chuang, *Quantum Computation and Quantum Information* (Cambridge University Press, Cambridge, United Kingdom, 2010).

[2] B. M. Terhal, Quantum error correction for quantum memories, Reviews of Modern Physics **87**, 307–346 (2015), arXiv:1302.3428.

[3] D. Aharonov and M. Ben-Or, in *Proceedings of the 29th Annual ACM Symposium on Theory of Computing* (Association for Computing Machinery, El Paso, TX, USA, 1997) pp. 176–188, arXiv:quant-ph/9611025.

[4] E. Knill, R. Laflamme, and W. H. Zurek, Resilient quantum computation: Error models and thresholds, Proceedings of the Royal Society of London. Series A: Mathematical, Physical and Engineering Sciences **454**, 365–384 (1998), arXiv:quant-ph/9702058.

[5] A. Y. Kitaev, Fault-tolerant quantum computation by anyons, Annals of Physics **303**, 2–30 (2003), arXiv:quant-ph/9707021.

[6] P. Aliferis, D. Gottesman, and J. Preskill, Quantum accuracy threshold for concatenated distance-3 codes, Quantum Information and Computation **6**, 97–165 (2006), arXiv:quant-ph/0504218.

[7] D. Aharonov, A. Kitaev, and J. Preskill, Fault-tolerant quantum computation with long-range correlated noise, Physical Review Letters **96**, 050504 (2006), arXiv:quant-ph/0510231.

[8] L. Egan, D. M. Debroy, C. Noel, A. Risinger, D. Zhu, D. Biswas, M. Newman, M. Li, K. R. Brown, M. Cetina, and C. Monroe, Fault-tolerant control of an error-corrected qubit, Nature **598**, 281–286 (2021), arXiv:2009.11482.

[9] S. Krinner, N. Lacroix, A. Remm, A. Di Paolo, E. Genois, C. Leroux, C. Hellings, S. Lazar, F. Swiadek, J. Herrmann, G. J. Norris, C. K. Andersen, M. Müller, A. Blais, C. Eichler, and A. Wallraff, Realizing repeated quantum error correction in a distance-three surface code, Nature **605**, 669–674 (2022), arXiv:2112.03708.

[10] Y. Zhao, Y. Ye, H.-L. Huang, Y. Zhang, D. Wu, H. Guan, Q. Zhu, Z. Wei, T. He, S. Cao, F. Chen, T.-H. Chung, H. Deng, D. Fan, M. Gong, C. Guo, S. Guo, L. Han, N. Li, S. Li, Y. Li, F. Liang, J. Lin, H. Qian, H. Rong, H. Su, L. Sun, S. Wang, Y. Wu, Y. Xu, C. Ying, J. Yu, C. Zha, K. Zhang, Y.-H. Huo, C.-Y. Lu, C.-Z. Peng, X. Zhu, and J.-W. Pan, Realization of an error-correcting surface code with superconducting qubits, Physical Review Letters **129**, 030501 (2022), arXiv:2112.13505.

[11] R. Acharya, I. Aleiner, R. Allen, T. I. Andersen, M. Ansmann, F. Arute, K. Arya, A. Asfaw, J. Atalaya, R. Babbush, D. Bacon, J. C. Bardin, J. Basso, A. Bengtsson, S. Boixo, G. Bortoli, A. Bourassa, J. Bovaird, L. Brill, M. Broughton, B. B. Buckley, D. A. Buell, T. Burger, B. Burkett, N. Bushnell, Y. Chen, Z. Chen, B. Chiaro, J. Cogan, R. Collins, P. Conner, W. Courtney, A. L. Crook, B. Curtin, D. M. Debroy, A. Del Toro Barba, S. Demura, A. Dunsworth, D. Eppens, C. Erickson, L. Faoro, E. Farhi, R. Fatemi, L. Flores Burgos, E. Forati, A. G. Fowler, B. Foxen, W. Giang, C. Gidney, D. Gilboa, M. Giustina, A. Grajales Dau, J. A. Gross, S. Habegger, M. C. Hamilton, M. P. Harrigan, S. D. Harrington, O. Higgott, J. Hilton, M. Hoffmann, S. Hong, T. Huang, A. Huff, W. J. Huggins, L. B. Ioffe, S. V. Isakov, J. Iveland, E. Jeffrey, Z. Jiang, C. Jones, P. Juhas, D. Kafri, K. Kechedzhi, J. Kelly, T. Khattar, M. Khezri, M. Kieferová, S. Kim, A. Kitaev, P. V. Klimov, A. R. Klots, A. N. Korotkov, F. Kostritsa, J. M. Kreikebaum, D. Landhuis, P. Laptev, K.-M. Lau, L. Laws, J. Lee, K. Lee, B. J. Lester, A. Lill, W. Liu, A. Locharla, E. Lucero, F. D. Malone, J. Marshall, O. Martin, J. R. McClean, T. McCourt, M. McEwen, A. Megrant, B. Meurer Costa, X. Mi, K. C. Miao, M. Mohseni, S. Montazeri, A. Morvan, E. Mount, W. Mruczkiewicz, O. Naaman, M. Neeley, C. Neill, A. Nersisyan, H. Neven, M. Newman, J. H. Ng, A. Nguyen, M. Nguyen, M. Y. Niu, T. E. O'Brien, A. Opremcak, J. Platt, A. Petukhov, R. Potter, L. P. Pryadko, C. Quintana, P. Roushan, N. C. Rubin, N. Saei, D. Sank, K. Sankaragomathi, K. J. Satzinger, H. F. Schurkus, C. Schuster, M. J. Shearn, A. Shorter, V. Shvarts, J. Skruzny, V. Smelyanskiy, W. C. Smith, G. Sterling, D. Strain, M. Szalay, A. Torres, G. Vidal, B. Villalonga, C. Vollgraff Heidweiller, T. White, C. Xing, Z. J. Yao, P. Yeh, J. Yoo, G. Young, A. Zalcman, Y. Zhang, and N. Zhu, Suppressing quantum errors by scaling a surface code logical qubit, Nature **614**, 676–681 (2023), arXiv:2207.06431.

[12] V. V. Sivak, A. Eickbusch, B. Royer, S. Singh, I. Tsioutsios, S. Ganjam, A. Miano, B. L. Brock, A. Z. Ding, L. Frunzio, S. M. Girvin, R. J. Schoelkopf, and M. H. Devoret, Real-time quantum error correction beyond break-even, Nature **616**, 50–55 (2023), arXiv:2211.09116.

[13] Y. Wang, S. Simsek, T. M. Gatterman, J. A. Gerber, K. Gilmore, D. Gresh, N. Hewitt, C. V. Horst, M. Matheny, T. Mengle, B. Neyenhuis, and B. Criger, Fault-tolerant one-bit addition with the smallest interesting colour code, Science Advances **10**, eado9024 (2024), arXiv:2309.09893.

[14] D. Bluvstein, S. J. Evered, A. A. Geim, S. H. Li, H. Zhou, T. Manovitz, S. Ebadi, M. Cain, M. Kalinowski, D. Hangleiter, J. P. Bonilla Ataides, N. Maskara, I. Cong, X. Gao, P. S. Rodriguez, T. Karolyshyn, G. Semeghini, M. J. Gullans, M. Greiner, V. Vuletic, and M. D. Lukin, Logical quantum processor based on reconfigurable atom arrays, Nature **626**, 58–65 (2024), arXiv:2312.03982.

[15] M. P. da Silva, C. Ryan-Anderson, J. M. Bello-Rivas, A. Chernoguzov, J. M. Dreiling, C. Foltz, F. Frachon, J. P. Gaebler, T. M. Gatterman, L. Grans-Samuelsson, D. Hayes, N. Hewitt, J. Johansen, D. Lucchetti, M. Mills, S. A. Moses, B. Neyenhuis, A. Paz, J. Pino, P. Siegfried, J. Strabley, A. Sundaram, D. Tom, S. J. Wernli, M. Zanner, R. P. Stutz, and K. M. Svore, Demonstration of logical qubits and repeated error correction with better-than-physical error rates, arXiv preprint (2024), arXiv:2404.02280.

[16] R. Acharya, L. Aghababaie-Beni, I. Aleiner, T. I. Andersen, M. Ansmann, F. Arute, K. Arya, A. Asfaw, N. Astrakhantsev, J. Atalaya, R. Babbush, D. Bacon, B. Ballard, J. C. Bardin, J. Bausch, A. Bengtsson, A. Bilmes, S. Blackwell, S. Boixo, G. Bortoli, A. Bourassa, J. Bovaird, L. Brill, M. Broughton, D. A. Browne, B. Buchea, B. B. Buckley, D. A. Buell, T. Burger, B. Burkett, N. Bushnell, A. Cabrera, J. Campero, H.-S. Chang, Y. Chen, Z. Chen, B. Chiaro, D. Chik, C. Chou, J. Claes, A. Y. Cleland, J. Cogan, R. Collins, P. Conner, W. Courtney, A. L. Crook, B. Curtin, S. Das, A. Davies, L. D. Lorenzo, D. M. Debroy, S. Demura, M. Devoret, A. D. Paolo, P. Donohoe, I. Drozdov, A. Dunsworth, C. Earle, T. Edlich, A. Eickbusch, A. M. Elbag, M. Elzouka, C. Erickson, L. Faoro, E. Farhi, V. S. Ferreira, L. F. Burgos, E. Forati, A. G. Fowler, B. Foxen, S. Ganjam, G. Garcia, R. Gasca, É. Genois, W. Giang, C. Gidney, D. Gilboa, R. Gosula, A. G. Dau, D. Graumann, A. Greene, J. A. Gross, S. Habegger, J. Hall, M. C. Hamilton, M. Hansen, M. P. Harrigan, S. D. Harrington, F. J. H. Heras, S. Heslin, P. Heu, O. Higgott, G. Hill, J. Hilton, G. Holland, S. Hong, H.-Y. Huang, A. Huff, W. J. Huggins, L. B. Ioffe, S. V. Isakov, J. Iveland, E. Jeffrey, Z. Jiang, C. Jones, S. Jordan, C. Joshi, P. Juhas, D. Kafri, H. Kang, A. H. Karamlou, K. Kechedzhi, J. Kelly, T. Khaire, T. Khattar, M. Khezri, S. Kim, P. V. Klimov, A. R. Klots, B. Kobrin, P. Kohli, A. N. Korotkov, F. Kostritsa, R. Kothari, B. Kozlovskii, J. M. Kreikebaum, V. D. Kurilovich, N. Lacroix, D. Landhuis, T. Lange-Dei, B. W. Langley, P. Laptev, K.-M. Lau, L. L. Guevel, J. Ledford, K. Lee, Y. D. Lensky, S. Leon, B. J. Lester, W. Y. Li, Y. Li, A. T. Lill, W. Liu, W. P. Livingston, A. Locharla, E. Lucero, D. Lundahl, A. Lunt, S. Madhuk, F. D. Malone, A. Maloney, S. Mandrá, L. S. Martin, S. Martin, O. Martin, C. Maxfield, J. R. McClean, M. McEwen, S. Meeks, A. Megrant, X. Mi, K. C. Miao, A. Mieszala, R. Molavi, S. Molina, S. Montazeri, A. Morvan, R. Movassagh, W. Mruczkiewicz, O. Naaman, M. Neeley, C. Neill, A. Nersisyan, H. Neven, M. Newman, J. H. Ng, A. Nguyen, M. Nguyen, C.-H. Ni, T. E. O'Brien, W. D. Oliver, A. Opremcak, K. Ottosson, A. Petukhov, A. Pizzuto, J. Platt, R. Potter, O. Pritchard, L. P. Pryadko, C. Quintana, G. Ramachandran, M. J. Reagor, D. M. Rhodes, G. Roberts, E. Rosenberg, E. Rosenfeld, P. Roushan, N. C. Rubin, N. Saei, D. Sank, K. Sankaragomathi, K. J. Satzinger, H. F. Schurkus, C. Schuster, A. W. Senior, M. J. Shearn, A. Shorter, N. Shutty, V. Shvarts, S. Singh, V. Sivak, J. Skruzny, S. Small, V. Smelyanskiy, W. C. Smith, R. D. Somma, S. Springer, G. Sterling, D. Strain, J. Suchard, A. Szasz, A. Sztein, D. Thor, A. Torres, M. M. Torunbalci, A. Vaishnav, J. Vargas, S. Vdovichev, G. Vidal, B. Villalonga, C. V. Heidweiller, S. Waltman, S. X. Wang, B. Ware, K. Weber, T. White, K. Wong, B. W. K. Woo, C. Xing, Z. J. Yao, P. Yeh, B. Ying, J. Yoo, N. Yosri, G. Young, A. Zalcman, Y. Zhang, N. Zhu, and N. Zobrist, Quantum error correction below the surface code threshold, Nature **638**, 920–926 (2025), arXiv:2408.13687.

[17] L. Caune, L. Skoric, N. S. Blunt, A. Ruban, J. McDaniel, J. A. Valery, A. D. Patterson, A. V. Gramolin, J. Majaniemi, K. M. Barnes, T. Bialas, O. Buğdaycı, O. Crawford, G. P. Gehér, H. Krovi, E. Matekole, C. Topal, S. Poletto, M. Bryant, K. Snyder, N. I. Gillespie, G. Jones, K. Johar, E. T. Campbell, and A. D. Hill, Demonstrating real-time and low-latency quantum error correction with superconducting qubits, arXiv preprint (2024), arXiv:2410.05202.

[18] B. W. Reichardt, A. Paetznick, D. Aasen, I. Basov, J. M. Bello-Rivas, P. Bonderson,

R. Chao, W. van Dam, M. B. Hastings, A. Paz, M. P. da Silva, A. Sundaram, K. M. Svore, A. Vaschillo, Z. Wang, M. Zanner, W. B. Cairncross, C.-A. Chen, D. Crow, H. Kim, J. M. Kindem, J. King, M. McDonald, M. A. Norcia, A. Ryou, M. Stone, L. Wadleigh, K. Barnes, P. Battaglino, T. C. Bohdanowicz, G. Booth, A. Brown, M. O. Brown, K. Cassella, R. Coxe, J. M. Epstein, M. Feldkamp, C. Griger, E. Halperin, A. Heinz, F. Hummel, M. Jaffe, A. M. W. Jones, E. Kapit, K. Kotru, J. Lauigan, M. Li, J. Marjanovic, E. Megidish, M. Meredith, R. Morshead, J. A. Muniz, S. Narayanaswami, C. Nishiguchi, T. Paule, K. A. Pawlak, K. L. Pudenz, D. R. Pérez, J. Simon, A. Smull, D. Stack, M. Urbanek, R. J. M. van de Veerdonk, Z. Vendeiro, R. T. Weverka, T. Wilkason, T.-Y. Wu, X. Xie, E. Zalys-Geller, X. Zhang, and B. J. Bloom, Logical computation demonstrated with a neutral atom quantum processor, arXiv preprint (2024), arXiv:2411.11822.

[19] J. Bausch, A. W. Senior, F. J. H. Heras, T. Edlich, A. Davies, M. Newman, C. Jones, K. Satzinger, M. Y. Niu, S. Blackwell, G. Holland, D. Kafri, J. Atalaya, C. Gidney, D. Hassabis, S. Boixo, H. Neven, and P. Kohli, Learning high-accuracy error decoding for quantum processors, Nature **635**, 834–840 (2024), arXiv:2310.05900.

[20] P. W. Shor, in *Proceedings of the 35th Annual Symposium on Foundations of Computer Science* (IEEE, Santa Fe, NM, USA, 1994) pp. 124–134.

[21] P. W. Shor, Polynomial-time algorithms for prime factorization and discrete logarithms on a quantum computer, SIAM Journal on Computing **26**, 1484–1509 (1997), arXiv:quant-ph/9508027.

[22] A. W. Harrow, A. Hassidim, and S. Lloyd, Quantum algorithm for solving linear systems of equations, Physical Review Letters **103**, 150502 (2009), arXiv:0811.3171.

[23] L. K. Grover, in *Proceedings of the 28th Annual ACM Symposium on Theory of Computing* (Association for Computing Machinery, Philadephia, PA, USA, 1996) arXiv:quant-ph/9605043.

[24] R. P. Feynman, Simulating physics with computers, International Journal of Theoretical Physics **21**, 467–488 (1982).

[25] S. Lloyd, Universal quantum simulators, Science **273**, 1073–1078 (1996).

[26] N. Bostrom, Are we living in a computer simulation?, Philosophical Quarterly **53**, 243–255 (2003).

[27] P. W. Shor, in *Proceedings of the 37th Annual Symposium on Foundations of Computer Science* (IEEE, Burlington, VT, USA, 1996) pp. 56–65, arXiv:quant-ph/9605011.

[28] D. Gottesman, in *Quantum Information Science and Its Contributions to Mathematics*, Proceedings of Symposia in Applied Mathematics, Vol. 68, edited by S. Lomonaco (American Mathematical Society, Providence, RI, USA, 2010) pp. 13–58, arXiv:0904.2557.

[29] P. Aliferis, in *Quantum Error Correction*, edited by D. A. Lidar and T. A. Brun (Cambridge University Press, Cambridge, United Kingdom, 2013) pp. 126–160.

[30] A. Calderbank, E. Rains, P. Shor, and N. Sloane, in *Proceedings of the IEEE International Symposium on Information Theory* (IEEE, Ulm, Germany, 1997) p. 292, arXiv:quant-ph/9608006.

[31] D. Gottesman, *Stabilizer Codes and Quantum Error Correction*, Ph.D. thesis, California Institute of Technology (1997), arXiv:quant-ph/9705052.

[32] S. Aaronson and D. Gottesman, Improved simulation of stabilizer circuits, Physical Review A **70**, 052328 (2004), arXiv:quant-ph/0406196.

[33] A. Y. Kitaev, Quantum computations: Algorithms and error correction, Russian Mathematical Surveys **52**, 1191 (1997).

[34] C. M. Dawson and M. A. Nielsen, The Solovay-Kitaev algorithm, Quantum Information and Computation **6**, 81–95 (2006), arXiv:quant-ph/0505030.

[35] S. T. Flammia and J. J. Wallman, Efficient estimation of Pauli channels, ACM Transactions on Quantum Computing **1**, 1–32 (2020), arXiv:1907.12976.

[36] E. Knill, Quantum computing with realistically noisy devices, Nature **434**, 39–44 (2005), arXiv:quant-ph/0410199.

[37] J. J. Wallman and J. Emerson, Noise tailoring for scalable quantum computation via randomized compiling, Physical Review A **94**, 052325 (2016), arXiv:1512.01098.

[38] S. J. Beale, J. J. Wallman, M. Gutiérrez, K. R. Brown, and R. Laflamme, Coherence in quantum error-correcting codes, Physical Review Letters **121**, 190501 (2018), arXiv:1805.08802.

[39] E. Huang, A. C. Doherty, and S. Flammia, Performance of quantum error correction with coherent errors, Physical Review A **99**, 022313 (2019), arXiv:1805.08227.

[40] J. K. Iverson and J. Preskill, Coherence in logical quantum channels, New Journal of Physics **22**, 073066 (2020), arXiv:1912.04319.

[41] D. McDuff and D. Salamon, in *Introduction to Symplectic Topology*, edited by D. McDuff and D. Salamon (Oxford University Press, New York, NY, USA, 2017) p. 0.

[42] M. Heinrich, *On stabiliser techniques and their application to simulation and certification of quantum devices*, Ph.D. thesis, University of Cologne (2021).

[43] S. Bravyi, M. Suchara, and A. Vargo, Efficient algorithms for maximum likelihood decoding in the surface code, Physical Review A **90**, 032326 (2014), arXiv:1405.4883.

[44] J. Edmonds, Paths, trees, and flowers, Canadian Journal of Mathematics **17**, 449–467 (1965).

[45] O. Higgott and C. Gidney, Sparse blossom: Correcting a million errors per core second with minimum-weight matching, Quantum **9**, 1600 (2025), arXiv:2303.15933.

[46] N. P. Breuckmann and J. N. Eberhardt, Quantum low-density parity-check codes, PRX Quantum **2**, 040101 (2021), arXiv:2103.06309.

[47] X.-G. Wen, Quantum orders in an exact soluble model, Physical Review Letters **90**, 016803 (2003), arXiv:quant-ph/0205004.

[48] J. P. Bonilla Ataides, D. K. Tuckett, S. D. Bartlett, S. T. Flammia, and B. J. Brown, The XZZX surface code, Nature Communications **12**, 2172 (2021), arXiv:2009.07851.

[49] S. B. Bravyi and A. Y. Kitaev, Quantum codes on a lattice with boundary, arXiv preprint (1998), arXiv:quant-ph/9811052.

[50] E. Dennis, A. Kitaev, A. Landahl, and J. Preskill, Topological quantum memory, Journal of Mathematical Physics **43**, 4452–4505 (2002), arXiv:quant-ph/0110143.

[51] A. G. Fowler, M. Mariantoni, J. M. Martinis, and A. N. Cleland, Surface codes: Towards practical large-scale quantum computation, Physical Review A **86**, 032324 (2012), arXiv:1208.0928.

[52] C. Chamberland, G. Zhu, T. J. Yoder, J. B. Hertzberg, and A. W. Cross, Topological and subsystem codes on low-degree graphs with flag qubits, Physical Review X **10**, 011022 (2020), arXiv:1907.09528.

[53] D. Poulin, Stabilizer formalism for operator quantum error correction, Physical Review Letters **95**, 230504 (2005), arXiv:quant-ph/0508131.

[54] N. Lacroix, A. Bourassa, F. J. H. Heras, L. M. Zhang, J. Bausch, A. W. Senior, T. Edlich, N. Shutty, V. Sivak, A. Bengtsson, M. McEwen, O. Higgott, D. Kafri, J. Claes, A. Morvan, Z. Chen, A. Zalcman, S. Madhuk, R. Acharya, L. A. Beni, G. Aigeldinger, R. Alcaraz, T. I. Andersen, M. Ansmann, F. Arute, K. Arya, A. Asfaw, J. Atalaya, R. Babbush, B. Ballard, J. C. Bardin, A. Bilmes, S. Blackwell, J. Bovaird, D. Bowers, L. Brill, M. Broughton, D. A. Browne, B. Buchea, B. B. Buckley, T. Burger, B. Burkett, N. Bushnell, A. Cabrera, J. Campero, H.-S. Chang, B. Chiaro, L.-Y. Chih, A. Y. Cleland, J. Cogan, R. Collins, P. Conner, W. Courtney, A. L. Crook, B. Curtin, S. Das, S. Demura, L. D. Lorenzo, A. D. Paolo, P. Donohoe, I. Drozdov, A. Dunsworth, A. Eickbusch, A. M. Elbag, M. Elzouka, C. Erickson, V. S. Ferreira, L. F. Burgos, E. Forati, A. G. Fowler, B. Foxen, S. Ganjam, G. Garcia, R. Gasca, É. Genois, W. Giang, D. Gilboa, R. Gosula, A. G. Dau, D. Graumann, A. Greene, J. A. Gross, T. Ha, S. Habegger, M. Hansen, M. P. Harrigan, S. D. Harrington, S. Heslin, P. Heu, R. Hiltermann, J. Hilton, S. Hong, H.-Y. Huang, A. Huff, W. J. Huggins, E. Jeffrey, Z. Jiang, X. Jin, C. Joshi, P. Juhas, A. Kabel, H. Kang, A. H. Karamlou, K. Kechedzhi, T. Khaire, T. Khattar, M. Khezri, S. Kim, P. V. Klimov, B. Kobrin, A. N. Korotkov, F. Kostritsa, J. M. Kreikebaum, V. D. Kurilovich, D. Landhuis, T. Lange-Dei, B. W. Langley, P. Laptev, K.-M. Lau, J. Ledford, K. Lee, B. J. Lester, L. L. Guevel, W. Y. Li, Y. Li, A. T. Lill, W. P. Livingston, A. Locharla, E. Lucero, D. Lundahl, A. Lunt, A. Maloney, S. Mandrà, L. S. Martin, O. Martin, C. Maxfield, J. R. McClean, S. Meeks, A. Megrant, K. C. Miao, R. Molavi, S. Molina, S. Montazeri, R. Movassagh, C. Neill, M. Newman, A. Nguyen, M. Nguyen, C.-H. Ni, M. Y. Niu, L. Oas, W. D. Oliver, R. Orosco, K. Ottosson, A. Pizzuto, R. Potter, O. Pritchard, C. Quintana, G. Ramachandran, M. J. Reagor, R. Resnick, D. M. Rhodes, G. Roberts, E. Rosenberg, E. Rosenfeld, E. Rossi, P. Roushan, K. Sankaragomathi, H. F. Schurkus, M. J. Shearn, A. Shorter, V. Shvarts, S. Small, W. C. Smith, S. Springer, G. Sterling, J. Suchard, A. Szasz,

A. Sztein, D. Thor, E. Tomita, A. Torres, M. M. Torunbalci, A. Vaishnav, J. Vargas, S. Vdovichev, G. Vidal, C. V. Heidweiller, S. Waltman, J. Waltz, S. X. Wang, B. Ware, T. Weidel, T. White, K. Wong, B. W. K. Woo, M. Woodson, C. Xing, Z. J. Yao, P. Yeh, B. Ying, J. Yoo, N. Yosri, G. Young, Y. Zhang, N. Zhu, N. Zobrist, H. Neven, P. Kohli, A. Davies, S. Boixo, J. Kelly, C. Jones, C. Gidney, and K. J. Satzinger, Scaling and logic in the color code on a superconducting quantum processor, arXiv preprint (2024), arXiv:2412.14256.

[55] A. Eickbusch, M. McEwen, V. Sivak, A. Bourassa, J. Atalaya, J. Claes, D. Kafri, C. Gidney, C. W. Warren, J. Gross, A. Opremcak, N. Z. K. C. Miao, G. Roberts, K. J. Satzinger, A. Bengtsson, M. Neeley, W. P. Livingston, A. Greene, Rajeev, Acharya, L. A. Beni, G. Aigeldinger, R. Alcaraz, T. I. Andersen, M. Ansmann, Frank, Arute, K. Arya, A. Asfaw, R. Babbush, B. Ballard, J. C. Bardin, A. Bilmes, Jenna, Bovaird, D. Bowers, L. Brill, M. Broughton, D. A. Browne, B. Buchea, B. B. Buckley, Tim, Burger, B. Burkett, N. Bushnell, A. Cabrera, J. Campero, H.-S. Chang, B. Chiaro, L.-Y. Chih, A. Y. Cleland, J. Cogan, R. Collins, P. Conner, W. Courtney, Alexander, L. Crook, B. Curtin, S. Das, A. D. T. Barba, S. Demura, L. D. Lorenzo, A. D. Paolo, P. Donohoe, I. K. Drozdov, A. Dunsworth, A. M. Elbag, M. Elzouka, C. Erickson, V. S. Ferreira, L. F. Burgos, E. Forati, A. G. Fowler, B. Foxen, S. Ganjam, Gonzalo, Garcia, R. Gasca, É. Genois, W. Giang, D. Gilboa, R. Gosula, A. G. Dau, Dietrich, Graumann, T. Ha, S. Habegger, M. Hansen, M. P. Harrigan, S. D. Harrington, S. Heslin, P. Heu, O. Higgott, R. Hiltermann, J. Hilton, H.-Y. Huang, A. Huff, W. J. Huggins, E. Jeffrey, Z. Jiang, X. Jin, C. Jones, C. Joshi, P. Juhas, A. Kabel, H. Kang, Amir, H. Karamlou, K. Kechedzhi, T. Khaire, T. Khattar, M. Khezri, S. Kim, B. Kobrin, A. N. Korotkov, F. Kostritsa, J. M. Kreikebaum, V. D. Kurilovich, D. Landhuis, Tiano, Lange-Dei, B. W. Langley, K.-M. Lau, J. Ledford, K. Lee, B. J. Lester, L. L. Guevel, Wing, Y. Li, A. T. Lill, A. Locharla, E. Lucero, D. Lundahl, A. Lunt, S. Madhuk, A. Maloney, S. Mandrà, L. S. Martin, O. Martin, C. Maxfield, J. R. McClean, S. Meeks, Anthony, Megrant, R. Molavi, S. Molina, S. Montazeri, R. Movassagh, M. Newman, A. Nguyen, M. Nguyen, C.-H. Ni, L. Oas, R. Orosco, K. Ottosson, A. Pizzuto, R. Potter, O. Pritchard, C. Quintana, G. Ramachandran, M. J. Reagor, D. M. Rhodes, E. Rosenberg, E. Rossi, K. Sankaragomathi, H. F. Schurkus, M. J. Shearn, A. Shorter, N. Shutty, V. Shvarts, S. Small, W. C. Smith, S. Springer, G. Sterling, J. Suchard, A. Szasz, A. Sztein, D. Thor, E. Tomita, A. Torres, M. M. Torunbalci, A. Vaishnav, J. Vargas, Sergey, Vdovichev, G. Vidal, C. V. Heidweiller, S. Waltman, J. Waltz, S. X. Wang, B. Ware, T. Weidel, T. White, K. Wong, B. W. K. Woo, M. Woodson, C. Xing, Z. J. Yao, P. Yeh, B. Ying, J. Yoo, N. Yosri, G. Young, A. Zalcman, Yaxing, Zhang, N. Zhu, S. Boixo, J. Kelly, V. Smelyanskiy, H. Neven, D. Bacon, Z. Chen, P. V. Klimov, P. Roushan, C. Neill, Y. Chen, and A. Morvan, Demonstrating dynamic surface codes, arXiv preprint (2024), arXiv:2412.14360.

[56] C. Gidney, Stim: A fast stabilizer circuit simulator, Quantum **5**, 497 (2021), arXiv:2103.02202.

[57] O. Higgott, PyMatching: A Python package for decoding quantum codes with minimum-weight perfect matching, ACM Transactions on Quantum Computing **3**, 16:1–16:16 (2022), arXiv:2105.13082.

[58] P.-J. H. S. Derks, A. Townsend-Teague, A. G. Burchards, and J. Eisert, Designing fault-tolerant circuits using detector error models, arXiv preprint (2024), arXiv:2407.13826.

[59] J. M. Martinis, Qubit metrology for building a fault-tolerant quantum computer, npj Quantum Information **1**, 1–3 (2015), arXiv:1510.01406.

[60] I. L. Chuang and M. A. Nielsen, Prescription for experimental determination of the dynamics of a quantum black box, Journal of Modern Optics **44**, 2455–2467 (1997), arXiv:quant-ph/9610001.

[61] H. Haeffner, W. Haensel, C. F. Roos, J. Benhelm, D. Chek-al-kar, M. Chwalla, T. Koerber, U. D. Rapol, M. Riebe, P. O. Schmidt, C. Becher, O. Gühne, W. Dür, and R. Blatt, Scalable multi-particle entanglement of trapped ions, Nature **438**, 643–646 (2005), arXiv:quant-ph/0603217.

[62] E. Nielsen, J. K. Gamble, K. Rudinger, T. Scholten, K. Young, and R. Blume-Kohout, Gate set tomography, Quantum **5**, 557 (2021), arXiv:2009.07301.

[63] D. Gross, Y.-K. Liu, S. T. Flammia, S. Becker, and J. Eisert, Quantum state tomography via compressed sensing, Physical Review Letters **105**, 150401 (2010), arXiv:0909.3304.

[64] H.-Y. Huang, R. Kueng, and J. Preskill, Predicting many properties of a quantum system from very few measurements, Nature Physics **16**, 1050–1057 (2020), arXiv:2002.08953.

[65] J. Emerson, R. Alicki, and K. Zyczkowski, Scalable noise estimation with random unitary operators, Journal of Optics B: Quantum and Semiclassical Optics **7**, S347–S352 (2005), arXiv:quant-ph/0503243.

[66] E. Magesan, J. M. Gambetta, and J. Emerson, Characterizing quantum gates via randomized benchmarking, Physical Review A **85**, 042311 (2012), arXiv:1109.6887.

[67] E. Magesan, J. M. Gambetta, B. R. Johnson, C. A. Ryan, J. M. Chow, S. T. Merkel, M. P. da Silva, G. A. Keefe, M. B. Rothwell, T. A. Ohki, M. B. Ketchen, and M. Steffen, Efficient measurement of quantum gate error by interleaved randomized benchmarking, Physical Review Letters **109**, 080505 (2012), arXiv:1203.4550.

[68] J. M. Gambetta, A. D. Corcoles, S. T. Merkel, B. R. Johnson, J. A. Smolin, J. M. Chow, C. A. Ryan, C. Rigetti, S. Poletto, T. A. Ohki, M. B. Ketchen, and M. Steffen, Characterization of addressability by simultaneous randomized benchmarking, Physical Review Letters **109**, 240504 (2012), arXiv:1204.6308.

[69] J. J. Wallman, Randomized benchmarking with gate-dependent noise, Quantum **2**, 47 (2018), arXiv:1703.09835.

[70] J. Helsen, X. Xue, L. M. K. Vandersypen, and S. Wehner, A new class of efficient randomized benchmarking protocols, npj Quantum Information **5**, 1–9 (2019), arXiv:1806.02048.

[71] R. Harper, S. T. Flammia, and J. J. Wallman, Efficient learning of quantum noise, Nature Physics **16**, 1184–1188 (2020), arXiv:1907.13022.

[72] R. Harper, W. Yu, and S. T. Flammia, Fast estimation of sparse quantum noise, PRX Quantum **2**, 010322 (2021), arXiv:2007.07901.

[73] S. T. Flammia and R. O'Donnell, Pauli error estimation via population recovery, Quantum **5**, 549 (2021), arXiv:2105.02885.

[74] S. T. Flammia, in *17th Conference on the Theory of Quantum Computation, Communication and Cryptography*, Vol. 232 (Schloss Dagstuhl, Urbana-Champaign, IL, USA, 2022) pp. 4:1–4:10, arXiv:2108.05803.

[75] A. Erhard, J. J. Wallman, L. Postler, M. Meth, R. Stricker, E. A. Martinez, P. Schindler, T. Monz, J. Emerson, and R. Blatt, Characterizing large-scale quantum computers via cycle benchmarking, Nature Communications **10**, 5347 (2019), arXiv:1902.08543.

[76] A. Carignan-Dugas, D. Dahlen, I. Hincks, E. Ospadov, S. J. Beale, S. Ferracin, J. Skanes-Norman, J. Emerson, and J. J. Wallman, The error reconstruction and compiled calibration of quantum computing cycles, arXiv preprint (2023), arXiv:2303.17714.

[77] N. Sundaresan, T. J. Yoder, Y. Kim, M. Li, E. H. Chen, G. Harper, T. Thorbeck, A. W. Cross, A. D. Córcoles, and M. Takita, Demonstrating multi-round subsystem quantum error correction using matching and maximum likelihood decoders, Nature Communications **14**, 2852 (2023), arXiv:2203.07205.

[78] S. Gicev, L. C. L. Hollenberg, and M. Usman, Quantum computer error structure probed by quantum error correction syndrome measurements, Physical Review Research **6**, 043249 (2024), arXiv:2310.12448.

[79] 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), arXiv:1708.08474.

[80] 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), arXiv:1812.08186.

[81] 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), arXiv:1907.02554.

[82] A. deMarti iOlius, J. E. Martinez, P. Fuentes, P. M. Crespo, and J. Garcia-Frias, Performance of surface codes in realistic quantum hardware, Physical Review A **106**, 062428 (2022), arXiv:2203.15695.

[83] O. Higgott, T. C. Bohdanowicz, A. Kubica, S. T. Flammia, and E. T. Campbell, Improved decoding of circuit noise and fragile boundaries of tailored surface codes, Physical Review X **13**, 031007 (2023), arXiv:2203.04948.

[84] K. Tiurev, P.-J. H. S. Derks, J. Roffe, J. Eisert, and J.-M. Reiner, Correcting non-independent and non-identically distributed errors with surface codes, Quantum **7**, 1123 (2023), arXiv:2208.02191.

[85] P. Aliferis and J. Preskill, Fault-tolerant quantum computation against biased noise, Physical Review A **78**, 052331 (2008), arXiv:0710.1301.

[86] E. Pelaez, V. Omole, P. Gokhale, R. Rines, K. N. Smith, M. A. Perlin, and A. Hashim, Average circuit eigenvalue sampling on NISQ devices, arXiv preprint (2024), arXiv:2403.12857.

[87] D. Litinski, Magic state distillation: Not as costly as you think, Quantum **3**, 205 (2019), arXiv:1905.06903.

[88] E. T. Hockings, QuantumACES.jl: Design noise characterisation experiments for quantum computers, Journal of Open Source Software **10**, 7707 (2025).

[89] L. Viola, E. Knill, and S. Lloyd, Dynamical decoupling of open quantum systems, Physical Review Letters **82**, 2417–2421 (1999), arXiv:quant-ph/9809071.

[90] M. Ware, G. Ribeill, D. Ristè, C. A. Ryan, B. Johnson, and M. P. da Silva, Experimental demonstration of Pauli-frame randomization on a superconducting qubit, Physical Review A **103**, 042604 (2021), arXiv:1803.01818.

[91] A. Hashim, R. K. Naik, A. Morvan, J.-L. Ville, B. Mitchell, J. M. Kreikebaum, M. Davis, E. Smith, C. Iancu, K. P. O'Brien, I. Hincks, J. J. Wallman, J. Emerson, and I. Siddiqi, Randomized compiling for scalable quantum computing on a noisy superconducting quantum processor, Physical Review X **11**, 041039 (2021), arXiv:2010.00215.

[92] A. Jain, P. Iyer, S. D. Bartlett, and J. Emerson, Improved quantum error correction with randomized compiling, Physical Review Research **5**, 033049 (2023), arXiv:2303.06846.

[93] S. T. Flammia and Y.-K. Liu, Direct fidelity estimation from few Pauli measurements, Physical Review Letters **106**, 230501 (2011), arXiv:1104.4695.

[94] O. Fawzi, A. Oufkir, and D. S. França, Lower bounds on learning Pauli channels with individual measurements, IEEE Transactions on Information Theory **71**, 2642–2661 (2025), arXiv:2301.09192.

[95] H.-Y. Huang, S. T. Flammia, and J. Preskill, Foundations for learning from noisy quantum experiments, arXiv preprint (2022), arXiv:2204.13691.

[96] S. Chen, Z. Zhang, L. Jiang, and S. T. Flammia, Efficient self-consistent learning of gate set Pauli noise, arXiv preprint (2024), arXiv:2410.03906.

[97] M. Held, P. Wolfe, and H. P. Crowder, Validation of subgradient optimization, Mathematical Programming **6**, 62–88 (1974).

[98] D. A. Bodenham and N. M. Adams, A comparison of efficient approximations for a weighted sum of chi-squared random variables, Statistics and Computing **26**, 917–928 (2016).

[99] J. P. Imhof, Computing the distribution of quadratic forms in normal variables, Biometrika **48**, 419–426 (1961).

[100] V. V. Fedorov, *Theory of Optimal Experiments*, edited by W. J. Studden and E. M. Klimko, Probability and Mathematical Statistics, Vol. 12 (Academic Press, New York, NY, USA, 1972).

[101] N. Fruitwala, A. Hashim, A. D. Rajagopala, Y. Xu, J. Hines, R. K. Naik, I. Siddiqi, K. Klymko, G. Huang, and K. Nowrouzi, Hardware-efficient randomized compiling, arXiv preprint (2024), arXiv:2406.13967.

[102] Y. E. Nesterov, A method of solving a convex programming problem with convergence rate $O(1/k^2)$, Soviet Mathematics Doklady **27**, 372–376 (1983).

[103] I. Sutskever, J. Martens, G. Dahl, and G. Hinton, in *Proceedings of the 30th International Conference on Machine Learning* (PMLR, Atlanta, GA, USA, 2013) pp. 1139–1147.

[104] G. Elfving, Optimum allocation in linear regression theory, The Annals of Mathematical Statistics **23**, 255–262 (1952).

[105] W. J. Studden, Elfving's theorem and optimal designs for quadratic loss, The Annals of Mathematical Statistics **42**, 1613–1621 (1971).

[106] S. J. Wright, Coordinate descent algorithms, Mathematical Programming **151**, 3–34 (2015).

[107] T. J. Mitchell, An algorithm for the construction of "D-optimal" experimental designs, Technometrics **16**, 203–210 (1974).

[108] J. Bezanson, A. Edelman, S. Karpinski, and V. B. Shah, Julia: A fresh approach to numerical computing, SIAM Review **59**, 65–98 (2017), arXiv:1411.1607.

[109] C. M. Hurvich and C.-L. Tsai, Regression and time series model selection in small samples, Biometrika **76**, 297–307 (1989).

[110] A. Gelman and J. Hill, in *Data Analysis Using Regression and Multilevel/Hierarchical Models*, Analytical Methods for Social Research (Cambridge University Press, Cambridge, United Kingdom, 2006) pp. 199–234.

[111] Z. Zhang, S. Chen, Y. Liu, and L. Jiang, A generalized cycle benchmarking algorithm for characterizing mid-circuit measurements, PRX Quantum **6**, 010310 (2025), arXiv:2406.02669.

[112] J. Hines and T. Proctor, Pauli noise learning for mid-circuit measurements, Physical Review Letters **134**, 020602 (2025), arXiv:2406.09299.

[113] D. Greenbaum, Introduction to quantum gate set tomography, arXiv preprint (2015), arXiv:1509.02921.

[114] A. Gelman and J. Hill, in *Data Analysis Using Regression and Multilevel/Hierarchical Models*, Analytical Methods for Social Research (Cambridge University Press, Cambridge, United Kingdom, 2006) pp. 387–414.

[115] S. Chen, Y. Liu, M. Otten, A. Seif, B. Fefferman, and L. Jiang, The learnability of Pauli noise, Nature Communications **14**, 52 (2023), arXiv:2206.06362.

[116] R. Harper, I. Hincks, C. Ferrie, S. T. Flammia, and J. J. Wallman, Statistical analysis of randomized benchmarking, Physical Review A **99**, 052350 (2019), arXiv:1901.00535.

[117] S. Laue, M. Mitterreiter, and J. Giesen, in *Advances in Neural Information Processing Systems*, Vol. 31 (Curran Associates, Montreal, Canada, 2018).

[118] S. Laue, M. Mitterreiter, and J. Giesen, A simple and efficient tensor calculus, Proceedings of the AAAI Conference on Artificial Intelligence **34**, 4527–4534 (2020), arXiv:2010.03313.

[119] J. Revels, M. Lubin, and T. Papamarkou, Forward-mode automatic differentiation in Julia, arXiv preprint (2016), arXiv:1607.07892.

[120] K. B. Petersen and M. S. Pedersen, *The Matrix Cookbook* (Technical University of Denmark, Copenhagen, Denmark, 2012).

[121] K. Mayer, A. Hall, T. Gatterman, S. K. Halit, K. Lee, J. Bohnet, D. Gresh, A. Hankin, K. Gilmore, J. Gerber, and J. Gaebler, Theory of mirror benchmarking and demonstration on a quantum computer, arXiv preprint (2023), arXiv:2108.10431.

[122] T. Proctor, S. Seritan, K. Rudinger, E. Nielsen, R. Blume-Kohout, and K. Young, Scalable randomized benchmarking of quantum computers using mirror circuits, Physical Review Letters **129**, 150502 (2022), arXiv:2112.09853.

[123] Z. Chen, K. J. Satzinger, J. Atalaya, A. N. Korotkov, A. Dunsworth, D. Sank, C. Quintana, M. McEwen, R. Barends, P. V. Klimov, S. Hong, C. Jones, A. Petukhov, D. Kafri, S. Demura, B. Burkett, C. Gidney, A. G. Fowler, A. Paler, H. Putterman, I. Aleiner, F. Arute, K. Arya, R. Babbush, J. C. Bardin, A. Bengtsson, A. Bourassa, M. Broughton, B. B. Buckley, D. A. Buell, N. Bushnell, B. Chiaro, R. Collins, W. Courtney, A. R. Derk, D. Eppens, C. Erickson, E. Farhi, B. Foxen, M. Giustina, A. Greene, J. A. Gross, M. P. Harrigan, S. D. Harrington, J. Hilton, A. Ho, T. Huang, W. J. Huggins, L. B. Ioffe, S. V. Isakov, E. Jeffrey, Z. Jiang, K. Kechedzhi, S. Kim, A. Kitaev, F. Kostritsa, D. Landhuis, P. Laptev, E. Lucero, O. Martin, J. R. McClean, T. McCourt, X. Mi, K. C. Miao, M. Mohseni, S. Montazeri, W. Mruczkiewicz, J. Mutus, O. Naaman, M. Neeley, C. Neill, M. Newman, M. Y. Niu, T. E. O'Brien, A. Opremcak, E. Ostby, B. Pató, N. Redd, P. Roushan, N. C. Rubin, V. Shvarts, D. Strain, M. Szalay, M. D. Trevithick, B. Villalonga, T. White, Z. J. Yao, P. Yeh, J. Yoo, A. Zalcman, H. Neven, S. Boixo, V. Smelyanskiy, Y. Chen, A. Megrant, and J. Kelly, Exponential suppression of bit or phase errors with cyclic error correction, Nature **595**, 383–387 (2021), arXiv:2102.06132.

[124] Y. Tomita and K. M. Svore, Low-distance surface codes under realistic quantum noise, Physical Review A **90**, 062320 (2014), arXiv:1404.3747.

[125] A. G. Fowler, Minimum weight perfect matching of fault-tolerant topological quantum error correction in average $O(1)$ parallel time, Quantum Information and Computation **15**, 145–158 (2015), arXiv:1307.1740.

[126] Y. Wu and L. Zhong, in *2023 IEEE International Conference on Quantum Computing and Engineering* (IEEE, Bellevue, WA, USA, 2023) pp. 928–938, arXiv:2305.08307.

[127] B. Barber, K. M. Barnes, T. Bialas, O. Buğdaycı, E. T. Campbell, N. I. Gillespie, K. Johar, R. Rajan, A. W. Richardson, L. Skoric, C. Topal, M. L. Turner, and A. B. Ziad, A real-time, scalable, fast and resource-efficient decoder for a quantum computer, Nature Electronics **8**, 84–91 (2025), arXiv:2309.05558.

[128] 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, Physical Review Letters **128**, 110504 (2022), arXiv:2110.04285.

[129] S. T. Spitz, B. Tarasinski, C. W. J. Beenakker, and T. E. O'Brien, Adaptive weight estimator for quantum error correction, Advanced Quantum Technologies **1**, 1800012 (2018), arXiv:1712.02360.

[130] T. Wagner, H. Kampermann, D. Bruß, and M. Kliesch, Pauli channels can be estimated from syndrome measurements in quantum error correction, Quantum **6**, 809 (2022), arXiv:2107.14252.

[131] H. Wang, P. Liu, Y. Liu, J. Gu, J. Baker, F. T. Chong, and S. Han, DGR: Tackling drifted and correlated noise in quantum error correction via decoding graph re-weighting, arXiv preprint (2024), arXiv:2311.16214.

[132] A. Remm, N. Lacroix, L. Bödeker, E. Genois, C. Hellings, F. Swiadek, G. J. Norris, C. Eichler, A. Blais, M. Müller, S. Krinner, and A. Wallraff, Experimentally informed decoding of stabilizer codes based on syndrome correlations, arXiv preprint (2025), arXiv:2502.17722.

[133] V. Sivak, M. Newman, and P. Klimov, Optimization of decoder priors for accurate quantum error correction, Physical Review Letters **133**, 150603 (2024), arXiv:2406.02700.

[134] E. T. Hockings, A. C. Doherty, and R. Harper, Scalable noise characterization of syndrome-extraction circuits with averaged circuit eigenvalue sampling, PRX Quantum **6**, 010334 (2025), arXiv:2404.06545.

[135] J. F. S. Miguel, D. J. Williamson, and B. J. Brown, A cellular automaton decoder for a noise-bias tailored color code, Quantum **7**, 940 (2023), arXiv:2203.16534.

[136] A. Dua, A. Kubica, L. Jiang, S. T. Flammia, and M. J. Gullans, Clifford-deformed surface codes, PRX Quantum **5**, 010347 (2024), arXiv:2201.07802.

[137] B. O'Donoghue, E. Chu, N. Parikh, and S. Boyd, Conic optimization via operator splitting and homogeneous self-dual embedding, Journal of Optimization Theory and Applications **169**, 1042–1068 (2016), arXiv:1312.3039.

[138] B. O'Donoghue, Operator splitting for a homogeneous embedding of the linear complementarity problem, SIAM Journal on Optimization **31**, 1999–2023 (2021), arXiv:2004.02177.

[139] A. Javadi-Abhari, M. Treinish, K. Krsulich, C. J. Wood, J. Lishman, J. Gacon, S. Martiel, P. D. Nation, L. S. Bishop, A. W. Cross, B. R. Johnson, and J. M. Gambetta, Quantum computing with Qiskit, arXiv preprint (2024), arXiv:2405.08810.

[140] E. T. Hockings, A. C. Doherty, and R. Harper, Improving error suppression with noise-aware decoding, arXiv preprint (2025), arXiv:2502.21044.

[141] J. Combes, K. V. Gulshen, M. P. Harrigan, P. J. Karalekas, M. P. da Silva, M. S. Alam, A. F. Brown, S. Caldwell, L. C. Capelluto, G. E. Crooks, D. Girshovich, B. R. Johnson, E. C. Peterson, A. M. Polloreno, N. C. Rubin, C. A. Ryan, A. N. Staley, N. A. Tezak, and J. A. Valery, Forest benchmarking: QCVV using PyQuil, Zenodo (2019).

[142] E. Nielsen, S. Seritan, T. Proctor, K. Rudinger, K. Young, A. Russo, R. Blume-Kohout, R. P. Kelly, J. K. Gamble, and L. Saldyt, pyGSTi, Zenodo (2022).

[143] S. J. Beale, K. Boone, A. Carignan-Dugas, A. Chytros, D. Dahlen, H. Dawkins, J. Emerson, S. Ferracin, V. Frey, I. Hincks, D. Hufnagel, P. Iyer, A. Jain, J. Kolbush, E. Ospadov, J. L. Pino, H. Qassim, J. Saunders, J. Skanes-Norman, A. Stasiuk, J. J. Wallman, A. Winick, and E. Wright, True-Q, Zenodo (2020).

[144] P. Nation, H. Paik, A. Cross, and Z. Nazario, The IBM Quantum heavy hex lattice (2021).

[145] G. P. Gehér, M. Jastrzebski, E. T. Campbell, and O. Crawford, To reset, or not to reset — that is the question, arXiv preprint (2024), arXiv:2408.00758.

[146] C. Gidney, Stability experiments: The overlooked dual of memory experiments, Quantum **6**, 786 (2022), arXiv:2204.13834.

[147] J. Helsen and M. Walter, Thrifty shadow estimation: Reusing quantum circuits and bounding tails, Physical Review Letters **131**, 240602 (2023), arXiv:2212.06240.

[148] M. B. Healy, R. Jokar, S. Thomas, V. R. Pascuzzi, K. Barton, T. A. Alexander, R. Elkabetz, B. C. Donovan, H. Horii, and M. Hillenbrand, Design and architecture of the IBM Quantum Engine Compiler, arXiv preprint (2024), arXiv:2408.06469.

[149] K. Rudinger, G. J. Ribeill, L. C. G. Govia, M. Ware, E. Nielsen, K. Young, T. A. Ohki, R. Blume-Kohout, and T. Proctor, Characterizing mid-circuit measurements on a superconducting qubit using gate set tomography, Physical Review Applied **17**, 014014 (2022), arXiv:2103.03008.

[150] A. Calzona, M. Papič, P. Figueroa-Romero, and A. Auer, Multi-layer cycle benchmarking for high-accuracy error characterization, arXiv preprint (2024), arXiv:2412.09332.